НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено
Завідувач кафедри
_____ О.В. Коваль
(підпис) (ініціали, прізвище)
“___”_____2020р.

ДИПЛОМНА РОБОТА

на здобуття ступеня бакалавра

з напряму підготовки
6.050101 “ Комп’ютерні науки та інформаційні технології “

на тему Автоматизована система налаштування та супроводу мультисервісних комп’ютерних мереж

Виконав (-ла): студент (-ка) __4__курсу, групи__ТР-62__

_____ Прокопченко Володимир Сергійович _____ _____
(прізвище, ім’я, по батькові) (підпис)

Керівник _____доцент Недашківський Олексій Леонідович_ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент ___ доцент кафедри ТЕС і АЕС к.т.н,
_____Побіровський Юрій Миколайович_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки 122 Комп’ютерні науки та інформаційні технології

Спеціалізація Геометричне моделювання в інформаційних системах

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр Коваль
(підпис)
”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту

Прокопченко Володимиру Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи «Автоматизована система налаштування та супроводу мультисервісних комп’ютерних мереж»

керівник роботи доц. Недашківський Олексій Леонідович
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”25” травня 2020р. № **1168-с**

2. Строк подання студентом роботи 15.06.20

3. Вихідні дані до роботи Система для автоматизації налуштування мережевих комутаторів. Побудована модель локальної мережі з розмежованним доступом за допомогою створенної системи автоматизації.

4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Ознайомлення із побудовою локальної мережі з розмежованим доступом, опис функцій необхідних для налаштування мережевого комутатора Cisco, демонстрація роботи розробленої системи налаштування комутаторів.

5. Перелік ілюстративного матеріалу
Ілюстрації до топологій, скріншоти робочого функціоналу створеної

системи, забраження створенної моделі локальної мережі, зображення результатів роботи програми.

6. Дата видачі завдання ”___”_____201___ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	16.11.19 - 10.01.20	
2.	Вивчення та аналіз задачі	11.01.20 - 15.03.20	
3.	Розробка архітектури та загальної структури системи	16.03.20 - 04.04.20	
4.	Розробка структур окремих підсистем	05.04.20 - 15.05.20	
5.	Програмна реалізація системи	16.05.20	
6.	Оформлення пояснювальної записки	20.05.20 - 29.05.20	
7.	Захист програмного продукту	09.06.20	
8.	Передзахист		
8.	Захист		

Студент _____ _____Прокопченко В.С._____
(підпис) (прізвище та ініціали,)

Керівник роботи _____ _Недашківський О.Л.___
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

У мережах інтернет-провайдерів в експлуатації знаходяться тисячі комутаторів різних моделей, на їх налаштування витрачається величезна кількість робочих годин. Зменшення втрачаемого часу на конфігурування мережевих комутаторів є актуальною задачею на сьогодні.

Метою роботи є створення системи автоматизації конфігурування мережевих комутаторів та продемонструвати її ефективність.

В результаті було розроблено систему автоматизації конфігурування мережевих комутаторів на основі розробленного програмного продукту, а для демонстрації ефективності побудовано модель локальної мережі з розмежованним доступом.

Пояснювальна записка містить 73 сторінок, 20 рисунків, 3 додатки.

ABSTRACT

Thousands of switches of various models are in operation in the networks of Internet providers, and a huge number of working hours are spent on their setup. Reducing the time lost in configuring network switches is an urgent task today.

The aim of the work is to create a system for automating the configuration of network switches and demonstrate its effectiveness.

As a result, a system for automating the configuration of network switches was developed on the basis of the developed software product, and a model of a local area network with limited access was built to demonstrate efficiency.

The explanatory note contains 73 pages, 20 figures, 3 appendic

Зміст

Перелік умовних позначень, скорочень і термінів

VLAN - Virtual Local Area Network

VTY – Virtual TelnetYpe

DHSP - Dynamic Host Configuration Protocol

STP - Spanning Tree Protocol

OSI - The Open Systems Interconnection model

SSH - Secure Shell

ВСТУП

Сьогодні нам важко уявити наше життя без інтернету. Перелічувати переваги інтернету і розповідати про нього можна дуже довго, тому ми маємо розуміти, що інтернет називають всесвітнім павутинням недарма, бо він складається з мільйонів локальних і глобальних приватних, публічних, академічних, ділових і урядових мереж, пов'язаних між собою з використанням різноманітних дротових і бездротових технологій.

В інформаційних системах зберігається, обробляється, циркулює різна інформація, втрата або спотворення якої може завдати істотної шкоди. Для уникнення таких ситуацій, важливо захистити дані від несанкціонованого доступу. Один з найбільш ефективних засобів захисту це розмежування доступу до інформаційних ресурсів.

Для захисту інформації використовують сегментування структури локальної обчислювальної мережі на окремі частини і визначення правил взаємодії цих частин один з одним. Для цього використовується технологія VLAN.

Основною перевагою технологій VLAN є надійність, оскільки VLAN налаштовується безпосередньо на мережевому обладнанні. Користувачі локальної мережі та особи, що намагаються отримати несанкціонований доступ до інформаційних ресурсів, не можуть отримати віддалений доступ до налаштування мережевого обладнання. Отримати доступ до налаштування мережевого обладнання можливо тільки через комп’ютер системного адміністратора, або ж фізично підключитись до мережевого обладнання через консольний кабель.

Саме на свічах проводиться сегментування локальної мережі. Свіч - перемикач, більш правильна назва мережевий комутатор, - це пристрій, що дозволяє з'єднувати кілька ділянок комп'ютерної мережі. Дане обладнання є свого роду багатопортовим мостом між комп'ютерами в мережі. Відмінною рисою комутатора є можливість передавати пакети даних конкретному одержувачу, що оптимізує роботу

мережі, знижуючи навантаження, підвищуючи безпеку.

На таких комутаторах будують свою мережу різні інтернет-провайдери, які надають доступ до інтернету майже в кожний будинок.
З розширенням мережі з’являється необхідність додавати комутаторам все більше нових функцій. Чим більше на комутаторах нових функцій, тим більше часу доводиться витрачати системним адміністраторам на проведення налаштування одного такого пристрою. У мережах інтернет-провайдерів в експлуатації знаходяться тисячі комутаторів різних моделей, на їх налаштування витрачається величезна кількість годин.

У цьому документі розповідається про систему, яка автоматизує процес налаштування свічів cisco. Буде надо пояснення необхідності розмежування доступу і як це реалізувати на комутаторі, а також пояснення про кожний окремий пункт функціоналу програмного забезпечення.

Розділ 1. Постановка задачі

1.1 Задача створення локальної мережі з розмежуванням доступу до інформаційних ресурсів.

Для демонстрації роботи системи автоматизації спершу необхідно побудувати локальну мережу.

За мету було поставлено створити локальну мережу для підприємства, яке складається із відділу бухгалтерії, відділу фінансів, відділу маркетингу та виробничого відділу. В мережі повинно бути 5 серверів, 2 з яких відповідно використовуються фінансовим та виробничим відділами, останні три є файловий сервер, веб сервер та поштовий сервер. Персонал з одного відділу не повинен мати доступу до серверів та комп'ютерів іншого відділу. У всіх відділів повинен бути доступ до файлового, поштового та веб серверів. В підприємстві повинен бути системний адміністратор котрій має доступ до мережевого обладнання, а також до комп'ютерів та серверів усіх відділів.

1.2 Задача автоматизації процесу налаштування комутаторів cisco

Для автоматизації процесу потрібно розробити програмний продукт за допомогою якого можна виконувати нижче наведенні пункти, не витрачаючи час на ввод кожної команди у інтерфейсі свіча.

1)Задати ім’я хоста

2)Виставити дату та час

2)Встановити пароль на режим конфігурації

4)Встановити пароль VTY

5) Підключити шифрування паролів

6) Налаштування STP

7)Створити VLAN

8)Додати індекс до влану

9)Додати опис до влану (description)

10)Додати IP комутатора у створенному влані

11)Вказати ідентифікатор моста(root priority)

12)Налаштувати на кожному порті швидкість

13)Налаштувати на кожному порті дюплекс

14)Вказати тип на кожному порті(клієнтський або транковий)

15)Додати порт до створенного влану

16)Вимкнути непотрібні порти(shutdown)

17)Налаштування на портах функції DHCP snooping trust

18) Налаштування на портах функції ARP Inspection trust 19)Налаштування на визначених портах технології PortFast

20)Налаштування на визначених портах технології RootGuard

21)Додати опис (description) на необхідних портах

Розділ 2. Технології вирішення поставлених задач

2.1 IP адресація

Третій рівень моделі OSI, або мережевий рівень, надає можливість різним пристроям передавати данні по мережі. Для реалізації цієї швидкісної передачі на вказаному рівні користюються чотирма процесами.

- Адресація крайніх пристроїв. Крайнім пристроям потрібно назначити спеціальну IP-адресу для того, щоб ідентифікувати їх у мережі.
- Маршрутизація. Рівень мережі дає можливість, яка дозволяє перенести пакети до кінцевого місця отримання в іншу мережу. Для передачі в інші мережі пакет має бути обраблений серез маршрутизатор. Мета маршрутизатора заключається у виборі ліній передачі для пакетів та маршрутизації їх до точки призначення. Цей процес називають маршрутизацією. Перш ніж досягти вузла призначення, пакети можуть пройти через декілька проміжних вузлів. Кожен маршрут на лінії передачі пакетів до точки призначення зветься стрибком.
- Інкапсуляція. Рівень мережі отримає одиницю даних протокола (PDU) від транспортного рівню. В цей час процессу, який зветься інкапсуляцією, рівень мережі присвоює дані заголовка-IP, таку як, IP-адресу джерела (відправника) та пункту призначення (одержувача).
- Деінсепсуляція. Коли пакети приходять до мережевого рівеня точки призначення, цей вузол аналізує IP заголовок пакета. Якщо IP адреса назначення в заголовку відповідає його власній IP адресі, IP-заголовок ліквідується з пакету. Після декапсуляції пакета вузлом мережі, отриманий PDU-рівень 4 пересилається до відповідної служби на транспортному рівні.

Протокол IP інкапсулює сегмент транспортного шару, додавши заголовок IP. Цей опис або заголовок використовують для передачі пакетів на вузол призначення. IP-заголовок буде присутній з момента надсилання пакетів з вихідного вузла до їх надходження до вузла призначення.

Протокол IP був розроблений як протокол з низьким навантаженням. Він надає лише ті функції, які необхідні для доставки пакета від вихідного вузла до вузла призначення через взаємопов'язану мережеву систему. Цей протокол не призначений для контролю та контролю потоку пакетів. При необхідності ці функції виконуються іншими протоколами на інших рівнях.

2.2 Технологія віртуальних мереж VLAN

VLAN (Virtual Local Area Network) - топологічналокальна комп'ютерна мережа, представляє керуючі групи із загальними списками вимог, котрі працюють так, як якщо б вони були налаштовані до широкомовного домену, незважаючи від їх фізичного місце знаходження. VLAN містить такі ж функції, які у фізичних локальних мережах, але дає дозвіл крайнім хостам групуватися, незважаючи на те, що вони не підключені до одної фізичної мережі. Такі групування можуть бути зроблені на основах програмних налаштувань замість переносу пристроїв до інших приміщень.

У комутованих мережах VLAN забезпечують адаптивність сегментації і організації. Мережі VLAN дозволяють згрупувати пристрої всередині локальної мережі. Група пристроїв в мережі VLAN взаємодіє так, ніби пристрої підключені за допомогою одного кабелю. Мережі VLAN грунтуються не через фізичні, а по логічних підключеннях.

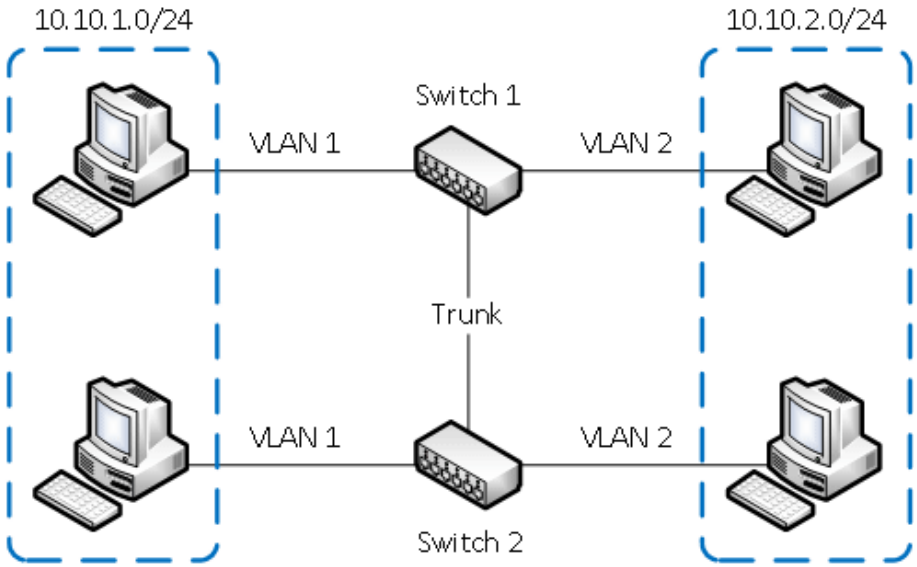

Рис 2.2 – поділ на влани

Мережі VLAN дозволяють адміністратору проводити сегментацію по функціям, проектним групам або областям застосування, незалежно від матеріального становища користувача або пристрою. Кожний VLAN вважається окремою логічною мережею. Пристрої в межах VLAN працюють таким чином, ніби

знаходяться у власній незалежній мережі, навіть якщо ділять одну загальну інфраструктуру з іншими VLAN. Будь-який комутаційний порт може належати мережі VLAN. Пакети одноадресної, широкомовної і під LGPL пересилаються і розсилаються тільки на кінцеві пристрої в межах вихідної мережі VLAN цих пакетів. Пакети, адресовані пристроїв, які не належать до VLAN, повинні пересилатися через пристрій, що підтримує маршрутизацію.

У комутованої мережі може бути кілька підмереж IP без використання декількох мереж VLAN. Однак пристрої будуть знаходитися в одному і тому ж домені широкомовної розсилки рівня 2. Це означає, що всі широкомовні розсилання рівня 2, наприклад ARP-запит, будуть прийматися всіма пристроями в комутованій мережі, навіть тими, які не призначені для прийому даної розсилки.

VLAN створює логічний широкомовний домен, який може включати декілька підмереж LAN. Поділяючи великі широкомовні домени на більш дрібні мережі, VLAN підвищують продуктивність мережі. Якщо пристрій в одній VLAN передає широкомовний кадр Ethernet, то цей кадр отримують всі пристрої в рамках цієї VLAN, пристрої ж в інших мережах VLAN цей кадр не отримують.

Мережі VLAN дозволяють реалізовувати політику забезпечення доступу і безпеки, враховуючи інтереси різних груп користувачів.

Продуктивність користувачів і адаптивність мережі відіграють важливу роль у процвітанні та успіху компанії. Мережі VLAN полегшують процес проектування мережі, що забезпечує допомогу у виконанні цілей організації. До основних переваг використання VLAN відносяться:

Безпека: групи, що володіють інформацією з обмеженим доступом, відокремлені від решти мережі, завдяки чому знижується вирогідність витоку секретної інформації. Як показано на малюнку, комп'ютери викладачів знаходяться в мережі VLAN 10 і повністю відокремлені від трафіку даних студентів і гостей.

Зниження витрат: завдяки економії на дорогих оновленнях мережевої інфраструктури і більш ефективному використанню наявної пропускної смуги і каналів відбувається зниження витрат.

Підвищення ефективності: розподіл однорідних мереж другого рівня на декілька окремих працюючих підгруп (широкомовного домену) знижує кількість непотрібного мережевого трафіка і збільшує ефективність.

Зменшення розміру доменів широкомовної розсилки: поділ мережі на підмережі VLAN знижує число приладів в домені широкомовної розсилки.

Підвищення продуктивності IT-відділу: мережі VLAN спрощують керування мережами, тому що користувач з аналогічними вимогами до мережі використовує одну і ту саму мережу VLAN. При введенні в експлуатацію нового комутатора на призначених портах реалізуються усі правила та методи, вже застосовані в цій конкретній VLAN. Також IT-фахівцям легше визначати функцію мережі VLAN, призначаючи їй відповідне ім'я.

Спрощене управління проектами та програмами: мережі VLAN об'єднують користувачів і мережеві пристрої для відповідності діловим або географічним вимогам мережі. Управління проектом і робота на прикладному рівні спрощені завдяки використанню поділу функцій.

Кожному VLAN в комутованій мережі відповідає IP-мережа. Таким чином, в проекті VLAN необхідно враховувати використання ієрархічної схеми мережевої адресації. Ієрархічна адресація передбачає впорядковане призначення номерів IP-мережі сегментам або мереж VLAN з урахуванням роботи мережі в цілому. Як показано на малюнку, блоки суміжних мережевих адрес резервуються і налаштовуються на пристроях в певній галузі мережі.

2.3 Топології комп'ютерних мереж

Всі комп'ютери в локальних мережах з'єднані лініями зв'язку. Такі звя’зки називають топологіями. Найбільш розповсюджені топології:

- Сітка(mesh)
- Зірка(Star)
- Шина(Bus)
- Кільце(Ring)

Сітка(mesh) - у топології кожен пристрій підключається до кожного іншого пристрою в мережі за допомогою спеціальної лінії "точка-точка". Коли ми кажемо, що лінія виділена, це означає, що посилання містить дані лише для двох підключених пристроїв. Скажімо, у нас є n пристроїв у мережі, то кожен пристрій повинен бути з'єднаний з (n-1) пристроями мережі. Кількість посилань у топології сітки n пристроїв складе n (n-1) / 2.

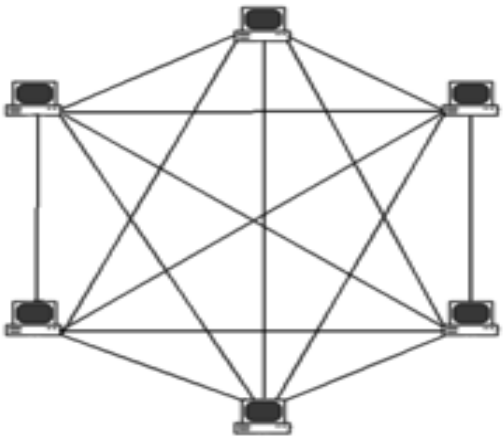

Рис. 2.3.1 – топологія сітка

Зірка(Star) - у зірковій топології кожен пристрій у мережі підключений до центрального пристрою, який називається концентратором. На відміну від топології Mesh, зіркова топологія не дозволяє здійснювати прямий зв'язок між пристроями, пристрій повинен зв’язуватися через концентратор. Якщо один пристрій хоче надіслати дані на інший пристрій, він повинен спочатку надіслати дані на концентратор, а потім концентратор передати ці дані на призначений пристрій.

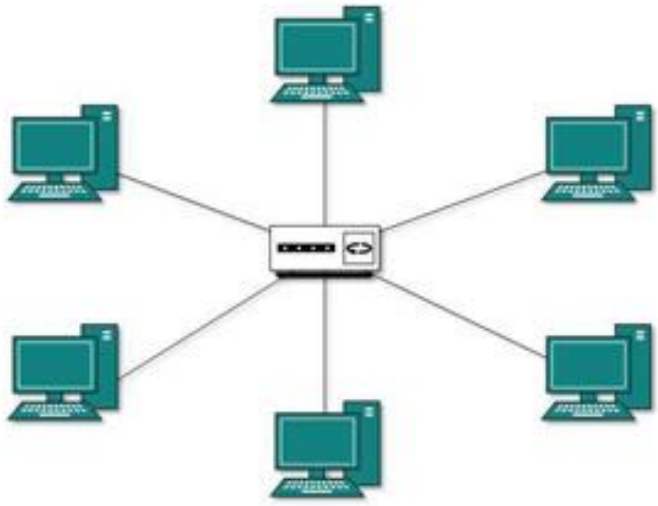

Рис. 2.3.2 – топологія зірка

Шина(Bus) - топології шини є основний кабель, і всі пристрої підключені до цього основного кабелю за допомогою окремих ліній. Є пристрій під назвою Т-конектор, який з'єднує допоміжні лінії до основного кабелю. Оскільки всі дані передаються по головному кабелю, існує обмеження допоміжних ліній та відстань, яку може мати головний кабель.

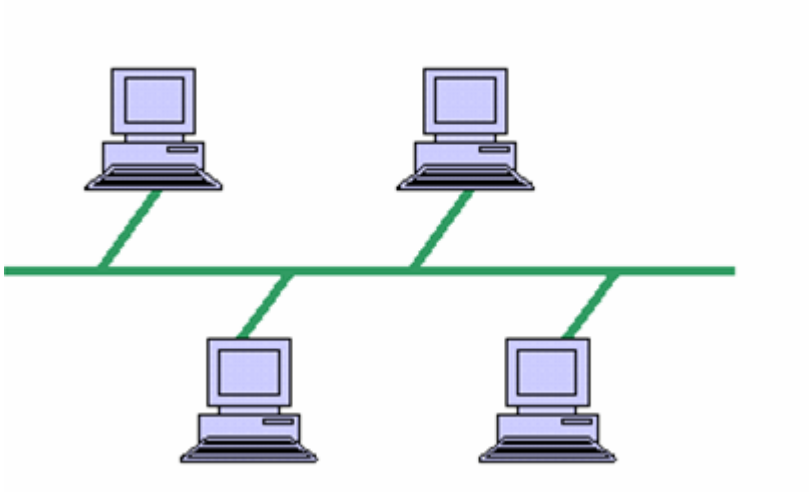

Рис. 2.3.3 – топологія шина

Кільце(Ring) - у топології кільця кожен пристрій з'єднано з двома пристроями з обох його боків. Існує два виділених між точкою та точковими зв’язками пристрою з пристроями по обидва боки від нього. Ця структура утворює кільце, тому вона

відома як топологія кільця. Якщо пристрій хоче надіслати дані на інший пристрій, то воно надсилає дані в одному напрямку, кожен пристрій в топології кільця має ретранслятор, якщо отримані дані призначені для іншого пристрою, то ретранслятор передає ці дані до тих пір, поки необхідний пристрій не отримає їх.

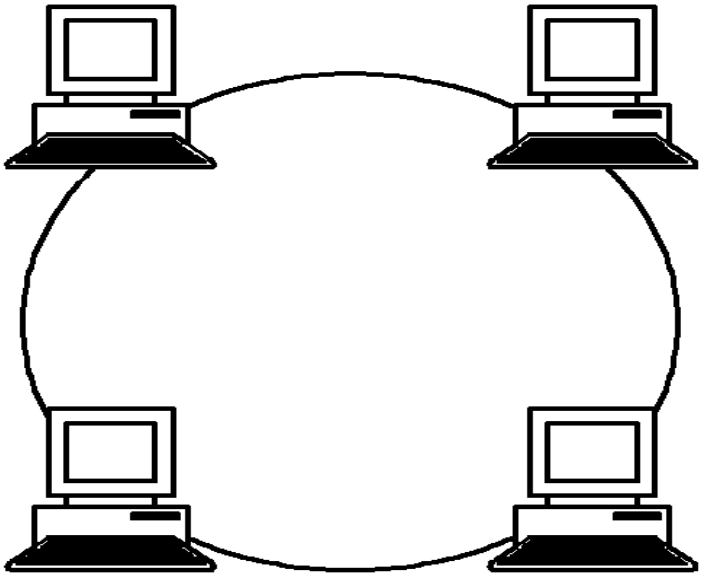

Рис. 2.3.4 – топологія кільце

2.4 Середовище передачі даних у комп'ютерних мережах

Середовище передачі даних - фізична субстанція, по якій відбувається передача тієї чи іншої інформації від джерела до приймача. Інформація переноситься за допомогою сигналів. Сигнали можуть мати різну природу:

- електричну (електрони по міді, заряджені іони);
- механічну (звукові хвилі по повітрю, сейсмічні хвилі в грунті);
- електромеханічну;
- електромагнітну (радіохвилі по повітрю, в безповітряному просторі або в грунті);
- оптичну (світло лазера по оптоволокну).

В комп’ютерних мережах використовують три основні середовища передачі данних:

- оптоволоконний кабель
- мідний кабель
- бездротове підключення

Хоча популярність бездротового підключення настільних комп'ютерів до мережі зростає, найбільш популярним середовищем передачі на фізичному рівні залишаються мідні й оптоволоконні кабелі.

Мідний кабель:

Мідні кабелі використовуються в мережах через їх невисоку вартість, простоту монтажу і низький електричний опір. Найбільш розповсюджене підключення між комутаторами та кінцевими пристроями, такими як роутери та комп’ютери. Однак при передачі сигналів по мідних кабелях можуть бути встановлені обмеження по дальності передачі і стійкості до перешкод.

Дані по мідних кабелях передаються у вигляді електричних імпульсів. Приймач в мережевому інтерфейсі цільового пристрою повинен отримати такий сигнал, який можна легко декодувати для відновлення відправленого сигналу. Однак чим більше дальність передачі сигналу, тим сильніше він спотворюється. Це

називається загасанням сигналів. Тому для всіх середовищ передачі даних на основі мідних кабелів в стандартах встановлені суворі обмеження на дальність передачі.

Оптоволоконні кабелі:

Оптоволоконні кабелі дозволяють передавати дані на великі відстані і з більш високою пропускною здатністю, ніж інші середовища передачі. На відміну від мідних проводів оптоволоконний кабель дозволяє передавати сигнали з більш низьким загасанням. Такий кабель також абсолютно несприйнятливий до впливу електромагнітних і радіочастотних перешкод. Оптичні кабелі зазвичай використовуються для з'єднання мережевих пристроїв один з одним.

Оптичне волокно - це гнучка, дуже тонка і прозора нитка з хімічно чистого скла товщиною трохи більше за людську волосину. Для передачі по оптоволоконному кабелю біти кодуються за допомогою світлових імпульсів. Оптоволоконний кабель діє як світловод, або «оптичний хвилевід», що забезпечує передачу світлового сигналу між двома кінцями кабелю з мінімальними втратами.

Оптоволокно складається з двох видів скляних компонентів (сердечника і внутрішньої оболонки) і захисної зовнішньої оболонки.

Хоча оптоволокно дуже тонке і погано переносить сильні вигини, але завдяки властивостям сердечника і оболонки воно дуже міцне. Завдяки своїй міцності оптичне волокно може використовуватися в найважчих умовах експлуатації.

2.5 Основний функціонал на комутаторах

Режими конфігурацій:

- Режим користувача

В цей режим ми потрапляємо спочатку, тут доступний тільки обмежений перелік команд, виконання яких не повинно зашкодити функціонуванню пристрою. Наприклад, з цього режиму можна подивитися версію операційної системи командою show version або запустити команду ping. Зазвичай доступ до цього режиму дають молодшим технікам, щоб вони могли діагностувати деякі проблеми самостійно, але не могли нічого зіпсувати в конфігурації.

- Привілейований режим

Для переходу в цей режим необхідно з призначеного для користувача режиму виконати команду enable і в разі необхідності ввести пароль. Після переходу, нам доступний повний перелік команд і можливість переходу в режим конфігурації без пароля. Таким чином, знаючи пароль на вхід на пристрій і пароль на привілейований режим, людина має повний доступ до комутатора, так як далі вже ніяких паролів вводити не потрібно. Для виходу назад в призначений для користувача режим використовується команда exit.

- Режим глобальної конфігурації

Цей режим дозволяє вносити зміни в конфігурацію пристрою. Для входу в нього необхідно з привілейованого режиму, виконати команду configure terminal. Введення паролів в даному випадку не буде потрібно.

- Режими специфічної конфігурації

Цих режимів безліч і вони є підрежимами основного режиму глобальної конфігурації. Наприклад, ввівши в режимі глобальної конфігурації команду interface FastEthernet 0/0 ми перейдемо в підрежим налаштування відповідного інтерфейсу (config-if). Безліч режимів специфічної конфігурації відповідає безлічі різних гілок глобальної конфігурації.

Безпека доступу на комутаторах:

Для захисту пристроїв cisco від несанкціонованого доступу використовується декілька видів паролів. Розглянемо налаштування паролів на консоль, паролів на підключення по telnet і ssh, а так же пароль для доступу в привілейований режим роботи пристрою. Паролі налаштовуються однаковим чином для маршрутизаторів і комутаторів.

- Пароль на консоль

При підключенні до пристрою через консольний кабель необхідно ввести пароль. За замовчуванням пароль на консоль відсутня. Треба розуміти, що фізична безпека пристрою найбільш важливий аспект захисту, так як маючи фізичний доступ до консольного порту, навіть не знаючи пароля його можна скинути.

- Пароль на Telnet і SSH

Доступ по протоколам telnet або ssh може бути здійснений тільки після того як на пристрої налаштований якась ip-адреса, а також задані паролі. У цьому важлива відмінність від доступу по консолі. Якщо паролі не задані, то по консолі можна зайти без пароля, а по Telnet або SSH зайти не можна - буде видано повідомлення, що поки немає пароля, віддалений вхід заборонений.

- Пароль на привілейований режим

Цей важливий пароль використовується для переходу з режиму користувача в привілейований. При вході на пристрій, незалежно від того, робимо ми це через VTY або через консоль, ми потрапляємо в призначений для користувача режим. Далі можна здійснити перехід в привілейований. Якщо встановлено пароль на привілейований режим, то його потрібно ввести, якщо не заданий - то все залежить від того способу, за яким ми підключилися до пристрою. При підключенні по консолі і відсутньому паролі на enable, перехід в привілейований режим відбудеться без введення пароля, якщо ж доступ здійснюється через Telnet або SSH, то без пароля на enable, нас в цей режим не пустять якщо пароль не заданий. З цієї причини початкова настройка маршрутизатора завжди проводиться через консоль і повинна включати в себе завдання всіх необхідних паролів.

- Служба шифрування паролів

У будь-якому випадку, паролі, задані для доступу по telnet або через консоль видно відкритим текстом. Для того, щоб приховати і ці паролі треба включити службу шифрування паролів, після чого всі паролі в файлі конфігурації, включаючи enable password, пароль на консоль і пароль на telnet, починають зберігатися в зашифрованому вигляді. Цю команду рекомендується завжди включати в базову настройку пристроїв cisco.

STP (Spanning Tree Protocol):

Мережевий протокол (або сімейство мережевих протоколів) призначений для автоматичного видалення циклів (петель комутації) з топології мережі на канальному рівні в Ethernet-мережах. В даний час протокол STP (або аналогічний) підтримується багатьма Ethernet-комутатори, як реальними, так і віртуальними, за винятком найпримітивніших.

Стани портів у STP:

1. Блокування (blocking)
2. Прослуховування (listening)
3. Навчання (learning)
4. Передача (forwarding)

Алгоритм дії STP (Spanning Tree Protocol)

- Після включення комутаторів в мережу, за замовчуванням кожен комутатор вважає себе кореневим (root).
- Кожен комутатор починає посилати по всіх портах конфігураційні Hello BPDU пакети раз в 2 секунди, максимальний проміжок 20 секунд.
- Якщо міст отримує BPDU з ідентифікатором моста (Bridge ID) меншим, ніж свій власний, він припиняє генерувати свої BPDU і починає ретранслювати BPDU з цим ідентифікатором. Таким чином в кінці кінців в цій мережі Ethernet залишається тільки один міст, який продовжує генерувати і передавати власні BPDU. Він і стає кореневим мостом (root bridge).
- Решта мости ретранслюють BPDU кореневого моста, додаючи в них власний

ідентифікатор і збільшуючи лічильник вартості шляху (path cost).

- Для кожного сегмента мережі, до якого приєднані два і більше портів мостів, відбувається визначення designated port - порту, через який BPDU, що приходять від кореневого моста, потрапляють в цей сегмент.
- Після цього всі порти в сегментах, до яких приєднані 2 і більше портів моста, блокуються за винятком root port і designated port.
- Кореневої міст продовжує посилати свої Hello BPDU раз в 2 секунди.

PVST та Rapid PVST

PVST (Per-VLAN Spanning Tree) - призначена для роботи в мережі з декількома VLAN. У PVST для кожного влану існує свій процес STP, що дозволяє незалежну і гнучке налаштування під потреби кожного Вланєв, але найголовніше, дозволяє використовувати балансування навантаження за рахунок того, що конкретний фізичний лінк може бути заблокований в одному Вланєв, але працювати в іншому.

RPVST - або як його ще називають у більш розгорнутому вигляді Rapid spanning tree protocol, по суті той же PVST але більш швидкий де час збіжності мить, ви втратите один пакет. При падінні одного лінка, час сходження між комутаторами буде 1 секунда.

Функція PortFast

Функція введена для уникнення проблем з підключенням до мережі. Ці проблеми можуть бути спричинені затримкою в портах з підтримкою STP, що переходять від стану блокування до стану передачі, після переходу з стану прослуховування та навчання. Порти з підтримкою STP, які підключені до таких пристроїв, як один комутатор, робоча станція або сервер, можуть отримати доступ до мережі лише після проходження всіх цих станів STP.

Деякі програми потребують негайного підключення до мережі, в іншому вони вимикаються. Увімкнення функції PortFast призводить до того, що комутатор або порт магістралі переходять у стан переадресації STP негайно або після з'єднання, минаючи таким чином стан прослуховування та навчання. Функція PortFast вмикається на рівні порту, і цей порт може бути або фізичним, або логічним портом.

Коли функція PortFast включена на комутаторі або порту магістралі, порт негайно переходить у стан пересилання STP. Хоча PortFast увімкнено, порт все ще бере участь у STP. Якщо порт виявляється частиною топології, яка може утворювати цикл, порт врешті переходить у режим блокування STP. PortFast зазвичай налаштовується на крайовому порту, а це означає, що порт не повинен отримувати ніяких STP BPDU. Якщо порт отримує будь-який STP BPDU, він переходить у звичайний / регулярний режим і братиме участь у станах прослуховування та навчання.

Root Guard

Це функція, що дозволяє запобігати появі шахрайських комутаторів і спуфинг. Ззахисний механізм в ситуаціях, коли вашої мережі і мережі вашого клієнта необхідно сформувати один домен STP, але ви хочете мати кореневої міст STP в своїй мережевий частини, і не хочете, щоб ваш клієнт взяв на себе вибір кореневого комутатора. У цих випадках ви повинні поставити Root Guard на портах у напрямку до клієнта.

DHCP Snoopind, Arp Inspection

Дані функції захищають вашу мережу від підміни DHCP сервера. На комутаторах вручну налаштовуються довірені порти, які як правило підключені до маршрутизатора або DHCP сервера.

Інша можливість це Dynamic Arp inspection. Теж захисна функція захищає від атаку типу Man-in-The-Middle. Це такий вид атаки, коли до вашої мережі підключається пристрій зловмисника і, наприклад, оголошує, що IP адреса, що належить авторизованому серверу, належить йому. Після цього всі дані, які відправляються на сервер переходять через пристрій зловмисника.

Налаштування портів

За замовчуванням, кожен порт налаштований таким чином, що пристрій сам визначає які налаштування на цьому порту використовувати, яку швидкість вибрати, який режим передачі даних. Така технологія називається Auto-negotiation (Автовизначення). Так само ці параметри можна задати «вручну», на кожному порту

пристрою.

Комутатори Cisco визначають автоматично швидкість між мережевими пристроями (наприклад між портом комутатора і мережевою картою комп'ютера), використовуючи деякі методи.

Якщо швидкості виставлені вручну і вони збігаються, то пристрої зможуть встановити з'єднання використовуючи електричні сигнали.

Якщо на комутаторі і на мережевому пристрої комп'ютера (для прикладу), встановлені вручну швидкості і вони не збігаються, то з'єднання не буде встановлено. Приблизно так само проходить і визначення режиму роботи з'єднання: half-duplex або full-duplex. Якщо обидва пристрої працюють в режимі автовизначення, і пристрої можуть працювати в duplex режимі, то цей режим і встановиться. Якщо на пристроях автовизначення вимкнено, то режим буде присвоєно за деякими правилами «за замовчуванням». Для 10 і 100 мегабітних інтерфейсів встановиться режим half-duplex, на 1000 мегабітних встановиться Full-Duplex. Для відключення автовизначення дуплексності необхідно вручну вказати настройки режиму. Ethernet пристрої можуть працювати в режимі Full-Duplex (FDX), тільки тоді, коли немає колізій в передавальної середовищі. Сучасні Ethernet технології говорять що колізії не відбуваються. Колізії відбуваються тільки там де є поділюване середовище передача даних, наприклад при топології шина, або при використанні такого пристрою як hub

Розділ 3. Аналіз літератури та розвитку мережевих комутаторів

Розглянемо модель OSI, це мережева модель стека мережевих протоколів. За допомогою даної моделі різні мережеві пристрої можуть з'єднуватися один з одним. Модель визначає різні рівні взаємодії систем. У моделі OSI засоби взаємодії діляться на сім рівнів: прикладний, уявлення, сеансовий, транспортний, мережевий, канальний і фізичний. Кожен рівень має справу з абсолютно певним аспектом взаємодії мережевих пристроїв.

Візьмемо перших три рівні для розуміння роботи комутатора.

Фізичний - на даному рівні визначаються властивості (механічні і оптичні / електричні) середовища передачі, що залежать від:

- типу кабелів і роз'ємів;
- розводки контактів в роз'ємах;
- схеми кодування сигналів для значень 0 і 1.

Канальним рівнем - забезпечуються створення, передача і прийом кадрів даних. Цим рівнем обслуговуються запити від вищого (мережевого) рівня, а для прийому-передачі пакетів використовується сервіс фізичного рівня. Згідно специфікаціям канальний рівень ділиться на наступні підрівні:

- LLC - управління логічним каналом (обслуговування мережевого рівня);
- MAC - управління доступом до середовища (доступ до фізичного середовища)

Мережевий рівень - відповідає за розподіл користувачів на групи. На мережевому рівні відбувається маршрутизація пакетів на основі перетворення MAC-адрес в мережеві адреси. Даним рівнем забезпечується прозора передача пакетів на транспортний рівень.

Нині існують і використовуються комутатори рівня L2 та L3, такі комутатори працюють на канальному і мережевому рівня відповідно.

По сучасним технологіям ми маємо можливість через інтерфейс комутаторів

змінювати конфігурацію, налаштовувати мережу як нам зручно. Комутатор передає дані лише безпосередньо отримувачу, це підвищує продуктивність і безпеку мережі, позбавляючи інші сегменти мережі від необхідності і можливості обробляти дані, які їм не призначалися. Проте до створення комутаторів існували інші технології.

Наприклад, повторювач (Repeater) - мережеве обладнання, призначене для збільшення відстані з'єднання з мережею і його розширення за межі одного сегмента або для організації двох гілок, шляхом повторення електричного сигналу «один в один». Бувають однопортові повторювачі і багатопортові. У термінах моделі OSI працює на фізичному рівні. Однією з перших завдань, яке стоїть перед будь-якою технологією транспортування даних, є можливість їх передачі на максимально велику відстань. Фізичне середовище накладає на цей процес своє обмеження - рано чи пізно потужність сигналу падає, і прийом стає неможливим.Звичне для аналогових систем посилення не годиться для високочастотних цифрових сигналів. Зрозуміло, при його використанні якийсь невеликий ефект може бути досягнутий, але зі збільшенням відстані спотворення швидко порушать цілісність даних. Проблема не нова, і в таких ситуаціях застосовують не посилення, а повторення сигналу. При цьому пристрій на вході має приймати сигнал, далі розпізнавати його первісний вигляд, і генерувати на виході його точну копію. Така схема в теорії може передавати дані на як завгодно великі відстані (якщо не враховувати особливості поділу фізичного середовища в Ethernet)

Проте Repeater не концентрує та не розподіляє сигнал, а лише передає його, тому трохи пізніше з’явились концентратори.

Концентратор (hub) працює на першому (фізичному) рівні мережевої моделі OSI, ретранслюючи вхідний сигнал з одного з портів в сигнал на всі інші (підключені) порти. Єдина перевага концентратора над комутатором - низька вартість - було актуально лише в перші роки розвитку мереж Ethernet. У міру вдосконалення і здешевлення електронних мікропроцесорних компонентів дана перевага концентратора повністю зійшло нанівець, так як вартість обчислювальної частини комутаторів і маршрутизаторів становить лише малу частку від вартості

роз'ємів, розділових трансформаторів, корпусу і блоку живлення, загальних для концентратора і комутатора.

Недоліки концентратора є логічним продовженням недоліків топології загальна шина, а саме - зниження пропускної здатності мережі у міру збільшення числа вузлів. Крім того, оскільки на фізичному рівні вузли не ізольовані один від одного, всі вони будуть працювати зі швидкістю передачі даних найгіршого вузла. Наприклад, якщо в мережі присутні вузли зі швидкістю 100 Мбіт / с і всього один вузол зі швидкістю 10 Мбіт / с, то всі вузли будуть працювати на швидкості 10 Мбіт / с, навіть якщо вузол 10 Мбіт / с взагалі не проявляє ніякої інформаційної активності. Ще одним недоліком є мовлення мережевого трафіку в усі порти, що знижує рівень мережевої безпеки. Мережевий концентратор також забезпечує безперебійну роботу мережі при відключенні пристрою від одного з портів або пошкодженні кабелю, на відміну, наприклад, від мережі на коаксіальному кабелі, яка в такому випадку припиняє роботу цілком.

Розділ 4. Засоби розробки

Для побудови мережі було використано середовище Cisco Packet Tracer - це багатофункціональна програма моделювання мереж, яка дозволяє експериментувати з поведінкою мережі і оцінювати можливі сценарії, налаштовувати (командами Cisco IOS) маршрутизатори і комутатори, взаємодіяти між декількома користувачами (через хмару).

У симуляторі реалізовані серії маршрутизаторів Cisco 800, 1800, 1900, 2600, 2800, 2900 і комутаторів Cisco Catalyst 2950, 2960, 3560, а також міжмережевий екран ASA 5505. Бездротові пристрої представлені маршрутизатором Linksys WRT300N, точками доступу і стільниковими вишками. Крім того є сервери DHCP, HTTP, TFTP, FTP, DNS, AAA, SYSLOG, NTP і EMAIL, робочі станції, різні модулі до комп'ютерів і маршрутизаторів, IP-фони, смартфони, хаби, а також хмара, що емулює WAN. Об'єднувати мережеві пристрої можна за допомогою різних типів кабелів, таких як прямі і зворотні пасивне, оптичні і коаксіальні кабелі, послідовні кабелі та телефонні пари.

Успішно дозволяє створювати навіть складні макети мереж, перевіряти на працездатність топологію мережі. Однак реалізована функціональність пристроїв обмежена і не надає всіх можливостей реального обладнання.

Для розробки системи автоматизації налаштувань комутаторів було використане середовище Microsoft Visual Studio 2019.

Інтегроване середовище розробки Visual Studio - це стартовий майданчик для написання, налагодження і складання коду, а також подальшої публікації додатків. Інтегроване середовище розробки (IDE) являє собою багатофункціональну програму, яку можна використовувати для різних аспектів розробки програмного забезпечення. Крім стандартного редактора і відладчика, які існують в більшості середовищ IDE, Visual Studio включає в себе компілятори, засоби автозавершения коду, графічні конструктори і багато інших функцій для спрощення процесу розробки. Середа Visual Studio доступна для Windows і Mac

Крім того в Visual Studio є можливість підключати та встановлювати різноманітні плагіни та інструменти. Платформа має функцію автозаповнення коду IntelliCode на базі II. Також є функція автоматичного виправлення на основі II IntelliSense, яка враховує стиль коду, необхідний різними компаніями, і дає індивідуальні рекомендації щодо його поліпшення. Система має загальну доступність функції спільної роботи в реальному часі Live Share. У Live Share, крім JavaScript, TypeScript і C #, є підтримка мов C ++ і Python.

Крім цього, Visual Studio 2019 має зручний інтерфейс для створення проектів, який значно прискорює написання і правку коду. Студія має пошук параметрів і команд, багато можливостей рефакторінга і функціональний відладчик. Присутня інтеграція з Git.

Для написання коду була використана мова C # - це мова програмування, що поєднує об'єктно-орієнтовані і контекстно-орієнтовані концепції. Розроблено в 1998-2001 роках групою інженерів під керівництвом Андерса Хейлсберга в компанії Microsoft як основна мова розробки додатків для платформи Microsoft .NET. Компілятор з C # входить в стандартну установку самої .NET, тому програми на ньому можна створювати і компілювати навіть без інструментальних засобів на кшталт Visual Studio.

C # відноситься до сім'ї мов з C-подібним синтаксисом, з них його синтаксис найбільш близький до C ++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато від своїх попередників - мов C ++, Delphi, Modula і Smalltalk - C #, спираючись на практику їх використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем: так, C # не підтримує множинне успадкування класів (на відміну від C ++) або виведення типів (на відміну від Haskell). C # розроблялася як мова програмування прикладного рівня для CLR і, як такий, залежить, перш за все, від можливостей самої CLR. Це стосується, перш за все, системи типів C #, яка відображає FCL. Присутність або

відсутність тих чи інших виразних особливостей мови диктується тим, чи може конкретна мовна особливість бути трансльований в відповідні конструкції CLR. Так, з розвитком CLR від версії 1.1 до 2.0 значно збагатився і сам C #; подібної взаємодії слід очікувати і в подальшому. (Однак ця закономірність була порушена з виходом C # 3.0, що представляє собою розширення мови, що не спираються на розширення платформи .NET.) CLR надає C #, як і всім іншим .NET-орієнтованим мовам, багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, прибирання сміття не реалізована в самому C #, а проводиться CLR для програм, написаних на C # точно так же, як це робиться для програм на VB.NET, J # і ін.

Розділ 5. Опис реалізації поставлених задач

5.1 Модель мережі в CiscoPacketTracer та використане обладнання

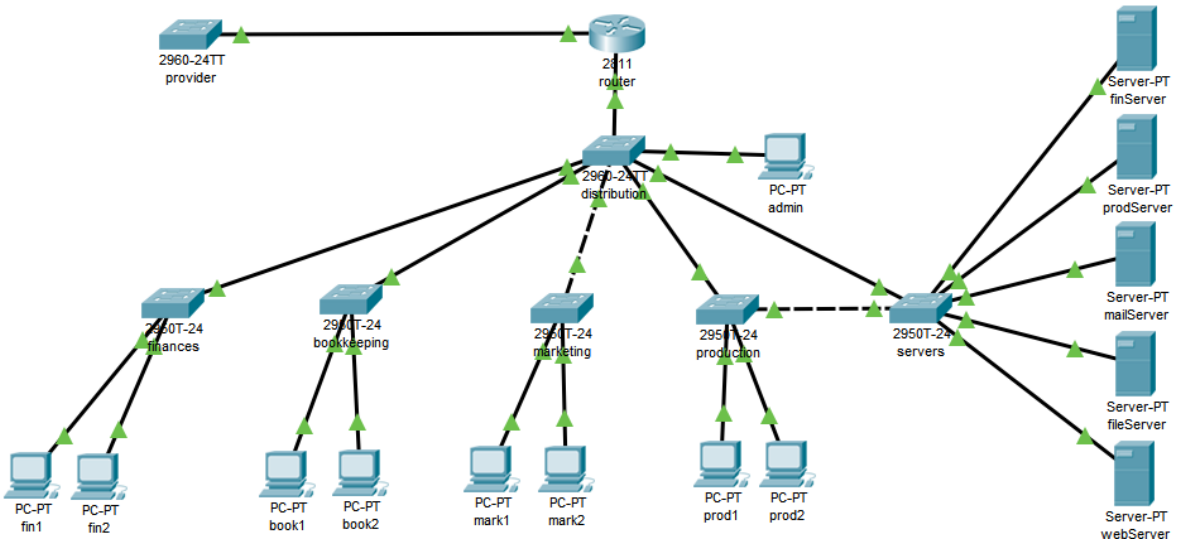

Рисунок 5.1 – Логічна схема мережі підприємства в програмі Packet Tracer.

Для побудови топології мережі, та перевірки її працездатності, використовується програмне забезпечення Cisco Packet Tracer.

В якості основного комутатору поширення було використано Cisco 2960-24TT, він має високі характеристики, що необхідно для комутатору поширення, адже через нього буде йти основний трафік. Також було використано 5 комутаторів доступу Cisco 2950, для чотирьох відділів та серверної. Усі прилади з'єднані витою парою.

Для доступу до інтернету, та можливості обміну даними між різними VLAN було використано маршрутизатор Cisco 2811.

Назва комутатору поширення – distribution, назви комутаторів доступу: finances, bookkeeping, marketing, production та servers відповідно. Назва маршрутизатору – router.

Сама мережа будується по ієрархічний топології, але між комутаторами production та servers існує додаткове з'єднання, котре не використовується при роботі мережі в нормальному режимі. Воно потрібно для перебудови мережі у разі пошкодження з'єднань між комутатором distribution та комутаторами servers, або production.

5.2 Вибір VLAN та IP-адрес для мережі

Для кожної групи пристроїв було обрано унікальний ідентифікатор VLAN.

В даної мережі було створено VLAN для відділів фінансів, бухгалтерії, маркетингу та виробничого відділу, для серверної, а також для адміністрування обладнання. Дані про VLAN даної мережі приведені в таблиці 4.1.

№ VLAN	**Ім'я VLAN**	**Примітка**
1	default	Не використовується
2	management	Для управління приладами
3	servers	Для серверної системи
4-100		Зарезервовано
101	finances	Для користувачів фінансового відділу
102	bookkeeping	Для користувачів відділу бухгалтерії
103	Marketing	Для користувачів відділу маркетингу
104	production	Для користувачів виробничого відділу
105	other	Для інших користувачів

Таблиця 4.1 - Планування VLAN

Для кожного з VLAN потрібно виділити окремий діапазон IP-адрес. Для даної мережі було обрано мережу 172.16.0.0/16, далі її потрібно поділити на декілька підмереж. Для підмереж окремих відділів було обрано префікс в 24 біти. Кожну підмережу потрібно сопоставити с певним VLAN, та зарезервувати IP-адреси в кожній підмережі для певних приладів. Інформація про IP адреси даної мережі знаходиться в таблиці 4.2.

IP адреса	**Примітка**	**VLAN**
172.16.0.0/16		
172.16.0.0/24	**Серверна**	**3**
172.16.0.1	Шлюз	
172.16.0.2	Поштовий сервер	
172.16.0.3	Файловий сервер	

172.16.0.4	Веб сервер	
172.16.0.5-254	Зарезервовані	
172.16.1.0/24	**Управління**	**2**
172.16.1.1	Шлюз	
172.16.1.2	Комутатор finances	
172.16.1.3	Комутатор bookkeeping	
172.16.1.4	Комутатор marketing	
172.16.1.5	Комутатор production	
172.16.1.6	Комутатор severs	
172.16.1.7	Комутатор distribution	
172.16.1.7-254	Зарезервовані	
172.16.2.0/24	**Відділ фінансів**	**101**
172.16.2.1	Шлюз	
172.16.2.2	Сервер відділу фінансів	
172.16.2.2-254	Пул доступних адрес для користувачів	
172.16.3.0/24	**Відділ бухгалтерії**	**102**
172.16.3.1	Шлюз	
172.16.32-254	Пул доступних адрес для користувачів	
172.16.4.0/24	**Відділ маркетингу**	**103**
172.16.4.1	Шлюз	
172.16.42-254	Пул доступних адрес для користувачів	
172.16.5.0/24	**Виробничий відділ**	**104**
172.16.5.1	Шлюз	
172.16.5.2	Сервер виробничого вибору	
172.16.52-254	Пул доступних адрес для користувачів	

Таблиця 4.2 - Планування IP-адрес

5.3 Налаштування маршрутизатора

За допомогою маршрутизатора буде здійснюватись доступ в інтернет , а також маршрутизація між різними VLAN мережами. Для забезпечення маршрутизації між декількома VLAN мережами, за допомогою тільки одного фізичного інтерфейсу маршрутизатора використовуються допоміжні інтерфейси.

Один допоміжний інтерфейс прив’язаний до конкретного фізичного інтерфейсу, та може оброблювати кадри з одної VLAN мережі. Для кожного VLAN корпоративної мережі потрібно створити свій допоміжний інтерфейс.

Створюється допоміжний інтерфейс командою int *фізичний інтерфейс*.*унікальний ідентифікатор* в режимі конфігурації терміналу. Унікальний ідентифікатор може бути вибраний за бажанням, але для зручності бажано використовувати ідентифікатор, що є однаковим з номером VLAN. Далі, в режимі конфігурації інтерфейсу потрібно прописати наступні команди:

description *текст* - задає назву інтерфейсу

encapsulation dot1Q *номер VLAN* - задає номер VLAN інтерфейсу

ip address *ip адреса* *маска під мережі* - ip інтерфейса в VLAN мережі

На даний момент усі мережі VLAN мають доступ до одна одної, згідно до завдання нам потрібно заборонити різним відділам доступ один до одного, цього можна добитися використовуючи ACL списки.

Потрібно створити 6 ACL списків для допоміжних інтерфейсів management, servers, finances, bookkeeping, marketing та production. Для інтерфейсів finances, bookkeeping, marketing та production потрібно створити ACL списки на трафік, що входить на інтерфейс, забороняючі пакети, що призначені для мереж інших відділів. Для інтерфейсу management потрібно створити ACL список на трафік, що виходить з інтерфейсу, дозволяючий тільки комп’ютеру адміністратора відправляти пакети. Для інтерфесу servers потрібно створити ACL список на трафік, що виходить з інтерфейсу, забороняючий увесь трафік, окрім трафіку з комп’ютеру адміністратору, а також дозволити доступ до поштового серверу по протоколам pop3 та smtp, до файлового серверу по протоколу ftp та до веб серверу по протоколу http.

На кожному з інтерфейсів також потрібно дозволити ICMP трафік.

ACL список створюється командою ip access-list extended *ім’я списку* в режимі конфігурації терміналу, правила створюються командами deny та permit, наступним зазначається протокол, далі зазначаються джерело та отримувач пакету, а також спеціальні умови.

Для додання ACL списку на інтерфейс, або допоміжний маршрутизатора використовується команда ip access-group *ім’я ACL списку* in/out в режимі конфігурації інтерфейсу. in буде означати, що правила вибраного ACL списку будуть примінятися на входячий трафік цього інтерфейсу, out буде означати, що правила вибраного ACL списку будуть примінятися на виходячий трафік.

Приклади команд:

permit ip any any eq ftp – пропускає пакети будь якого протоколу, які проходять по порту FTP.

deny icmp 172.16.0.0 0.0.0.255 any – дозволяє пакети протоколу icmp, в яких адресою джерела є адреса з мережі 172.16.0.0/24, маска під мережі в правилах ACL записується в вайлдкард форматі, тобто в зворотньому вигляді – 255.255.255.255 – 255.255.255.0 = 0.0.0.255

5.4 Створення системи для автоматизації налаштування комутаторів cisco

У середовищі Visual studio було розроблено програму для автоматизації внесення конфігурації на свічі cisco. Основний принцип роботи програми полягає у зчитуванні вхідних данних з полів, які заповнює користувач, після чого переклад отриманних данних у відповідні команди для налаштування комутатора. Всі команди записуються у вихідний txt документ. Для швидкого налаштування, потрібно перенести конфігурацію з документа на інтерфейс комутатора.

Пояснення до списку команд у вихідному документі:

Спершу у вихідний документ записується команда входу и привілейований режим і вносяться данні по часу та даті:

- ✓ enable
- ✓ clock set “H., M., S., Day, Month, Year”

Запис команди входу в режим конфігурації:

- ✓ configure terminal

Запис імені хоста обранного користувачем:

- ✓ hostname “ім’я хоста”

Запис підключення обраного користувачем STP протоколу на вибіо одного з двух “PVST” або “Rapaid PVST”:

- ✓ spanning-tree mode pvst

  або
- ✓ spanning-tree mode rapid-pvst

Запис підключення шифрування паролів, якщо користувач обрав цей пункт:

- ✓ service password-encryption

Запис підключення паролю на консоль обранного користувачем:

- ✓ line console 0
- ✓ password “пароль”
- ✓ login
- ✓ exit

Запис підключення паролю VTY обранного користувачем:

- ✓ line vty 0 15
- ✓ password “пароль”
- ✓ login
- ✓ exit

Запис підключення паролю на привілейований режим обранного користувачем:

- ✓ enable secret “пароль”

Запис створення стандартного 1-ого влану

- ✓ intarface vlan 1
- ✓ exit

Запис створення VLAN з данними які обрав користувач (наведені нижче команди будуть повторюватись для кожного створенного влану)

- ✓ int vlan “номер влану вказаний користувачем”
- ✓ description “опис влану вказаний користувачем”
- ✓ ip address “ip комутатора у данному влані, вказується користувачем” 255.255.255.0
- ✓ exit
- ✓ spanning-tree vlan “номер влану” root primary
- ✓ spanning-tree vlan “номер влану” priority “ номер root пріорітет обраний користувачем”

Записи налаштування кожного порта в залежності від вказаних користувачем данних (наведені нижче команди будуть повторюватись в залежності від кількості портів):

- ✓ interface “номер порта”
- ✓ speed “обрана користувачем швидкість”
- ✓ duplex “обраний користувачем дюплекс”
- ✓ description “вказаний користувачем опис порта”
- ✓ (якщо користувач обрав цей пункт) spanning-tree portfast
- ✓ (якщо користувач обрав цей пункт) spanning-tree guard root
- ✓ (якщо користувач обрав цей пункт) ip dhcp snooping trust

- ✓ (якщо користувач обрав цей пункт) ip arp inspection trust
- ✓ (складений або піднятий порт в залежності від вибору користувача) no shutdown
- ✓ switchport mode trunk

  або (в залежності від обранного користувачем типу порта)
- ✓ switchport mode access
- ✓ (якщо транк)switchport trunk allowed vlan “номера вланів”

  або
- ✓ (якщо access)switchport access vlan
- ✓ exit

Запис виходу з режиму конфігурації та збереження змін на комутаторі:

- ✓ exit
- ✓ write

Поясненння додаткового функціоналу створенної програми

Для зручності користувача у створені налаштувань для комутатора, було додано наступний функціонал:

- створення одразу декількох вланів при цьому є можливість відредагувати або видалити вже створенний влан
- Випадаючі вікна з можливостю обрати в них один конкретний пункт зі списку запропонованних
- При виборі дати надається можливість автоматично вказати поточну дату з комп’ютера
- Доданна функція вибору моделі комутатора зі списка доступних комутаторів для створення конфігурації(у списку реалізована тільки модель комутатора cisco для демонстрації, в майбутніх оновленнях будуть додаватись нові моделі)

5.5 Налаштування комутаторів

Завдяки створеній системі, робимо швидке налаштування комутаторів.

Для налаштування VLAN на комутаторах потрібно кожному з використовуваних портів призначити режим роботи Access, або Trunk. Режим Access використовується на портах, до яких підключені кінцеві прилади, на Access портах може бути налаштований тільки один VLAN. Режим Trunk використовується для з'єднання між комутаторами, на Trunk портах може бути налаштовано декілька VLAN. Інформацію про налаштування портів комутаторів знаходиться в таблиці 4.3.

Ім'я приладу	**Порт**	**Назва**	**VLAN**	
			Access	**Trunk**
distribution	Fa0/1	finances		2, 101
	Fa0/2	bookkeeping		2, 102
	Fa0/3	marketing		2, 103
	Fa0/4	production		2, 104
	Fa0/5	admin	2	
	Gig0/1	servers		2, 3, 101, 104
	Gig0/2	router		2, 3, 101-105
finances	Fa0/1	fin1	101	
	Fa0/2	fin2	101	
	Fa0/24	distribution		2, 101
bookkeeping	Fa0/1	book1	102	
	Fa0/2	book2	102	
	Fa0/24	distribution		2, 102
marketing	Fa0/1	mark1	103	
	Fa0/2	mark2	103	
	Fa0/24	distribution		2, 103
production	Fa0/1	prod1	104	
	Fa0/2	prod2	104	

	Fa0/24	distribution		2, 3, 101, 104
	Gig0/1	serversReserve		2, 3, 101, 104
servers	Fa0/1	finServer	101	
	Fa0/2	prodServer	104	
	Fa0/3	mailServer	3	
	Fa0/4	fileServer	3	
	Fa0/5	webServer	3	
	Gig0/1	distribution		2, 3, 101, 104
	Gig0/1	distributionReserve		2, 3, 101, 104

Таблиця 4.3 - Планування VLAN на портах комутаторів

Для ефективного використання мережі резервним з'єднанням потрібно зробити саме з'єднання між комутаторами production та servers, щоб комутатори production та servers мали пряме підключення до комутатору distribution.

5.6 Перевірка працездатності мережі

Використовуючи команду ping, можна перевірити чи є з'єднання між приладами в кожному з відділів, а також чи має кожен відділ з'єднання з серверами. Як показано на рис. 5.6.1-5.6.5, з'єднання між приладами в одному відділі, та між відділами та серверами присутнє.

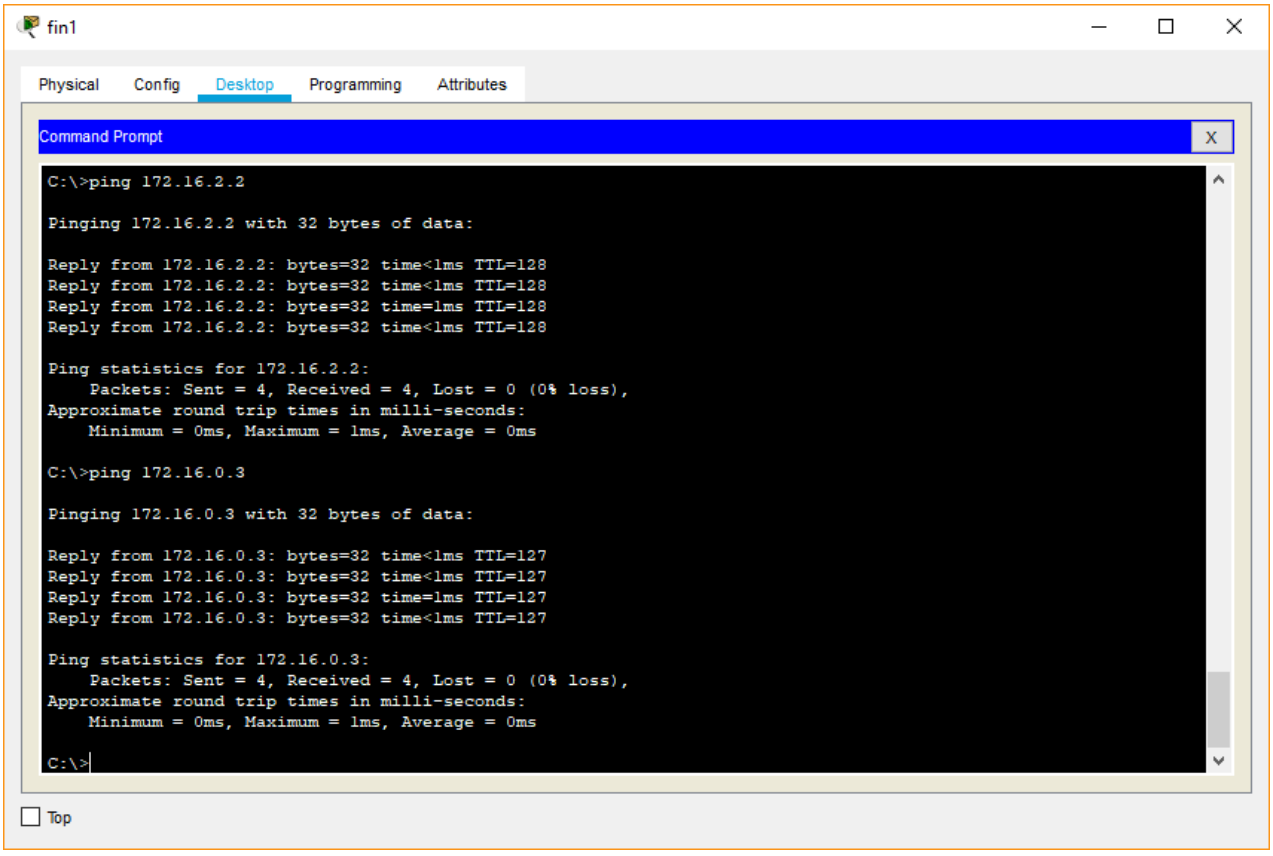

Рис. 5.6.1 – Результат команд ping, на комп’ютері fin1

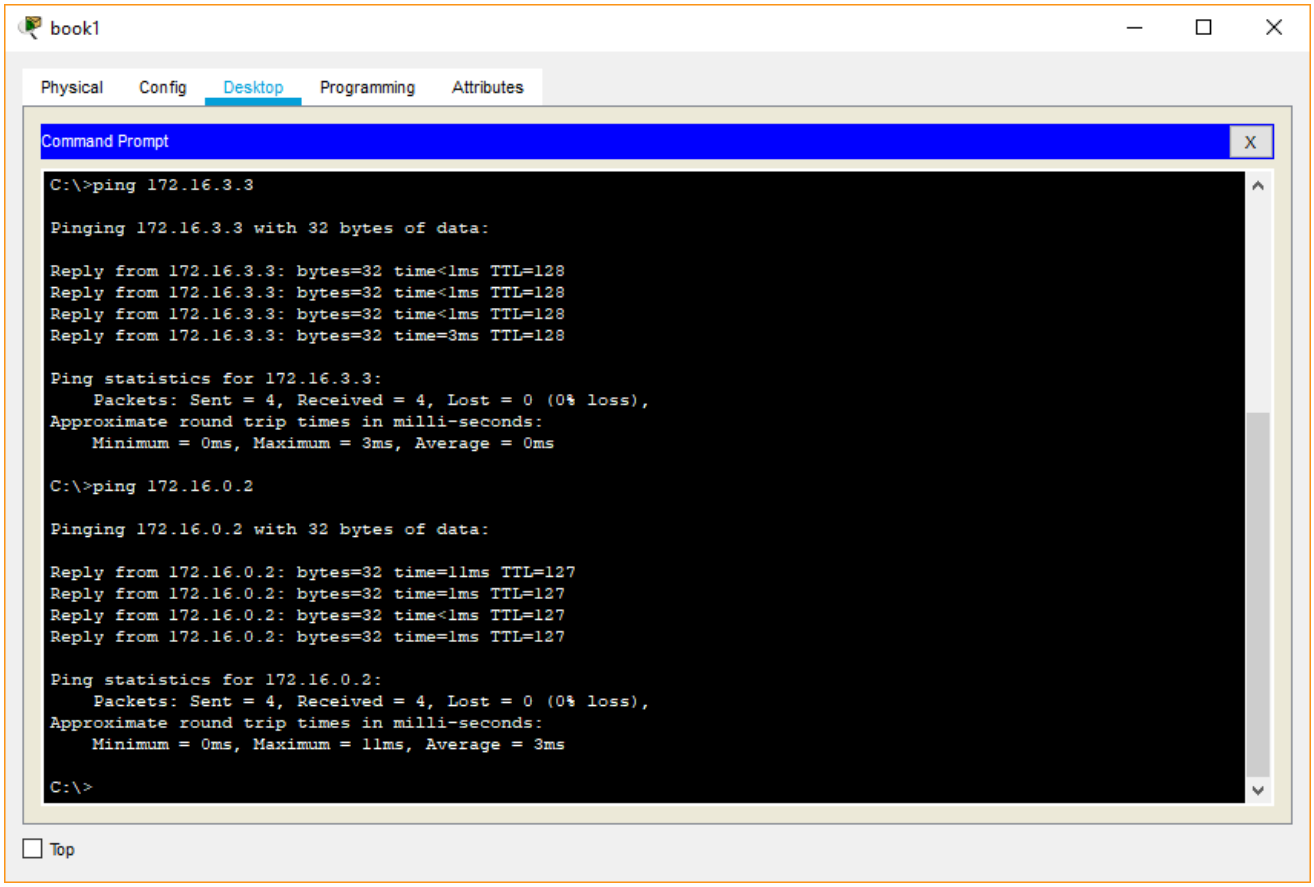

Рис. 5.6.2 – Результат команд ping, на комп’ютері book1

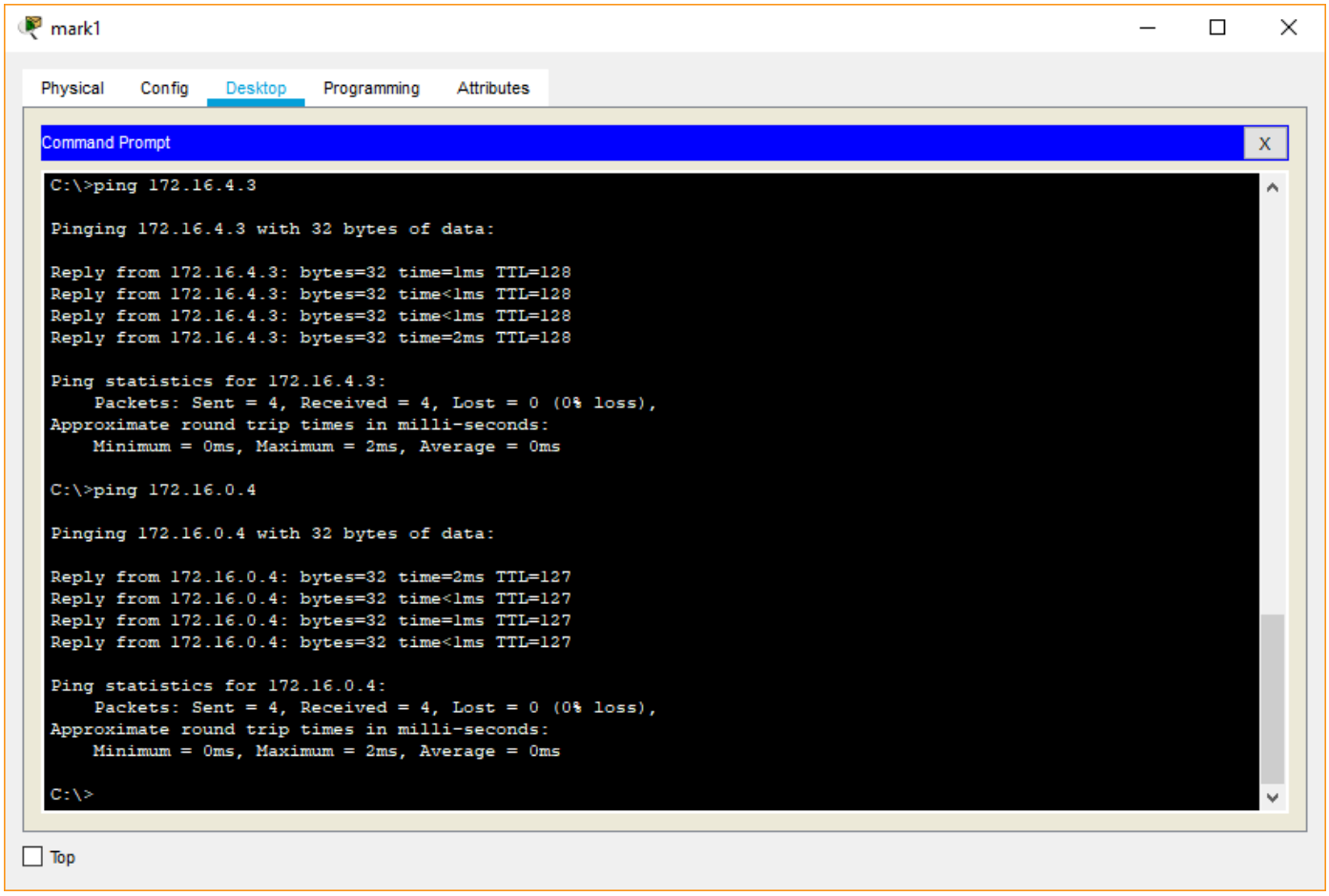

Рис. 5.6.3 – Результат команд ping, на комп’ютері mark1

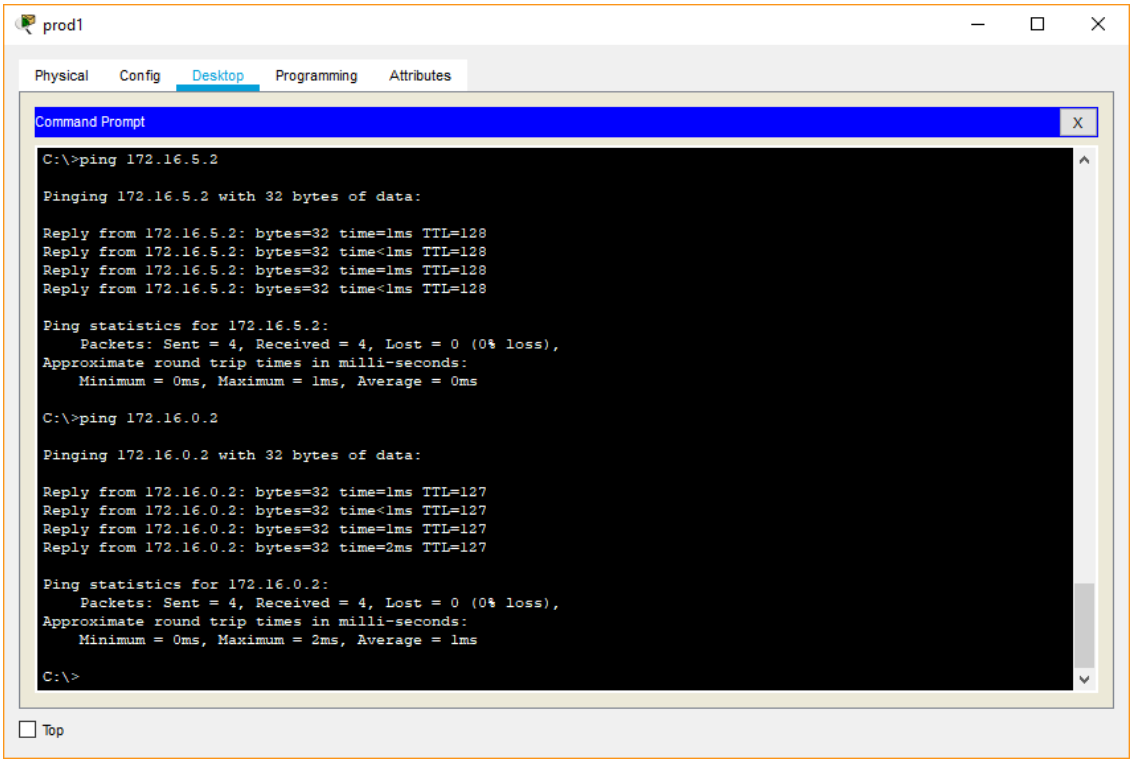

Рис. 5.6.4– Результат команд ping, на комп’ютері prod1

Далі потрібно перевірити можливість віддаленого налаштування комутаторів та маршрутизатору за допомогою Telnet, з комп'ютеру адміністратору. Для цього використовуємо команду telnet в командній строці комп'ютеру admin. Як показано на рис. 12, комп'ютер адміністратора має доступ до маршрутизатору, а також до комутаторів finances, та marketing.

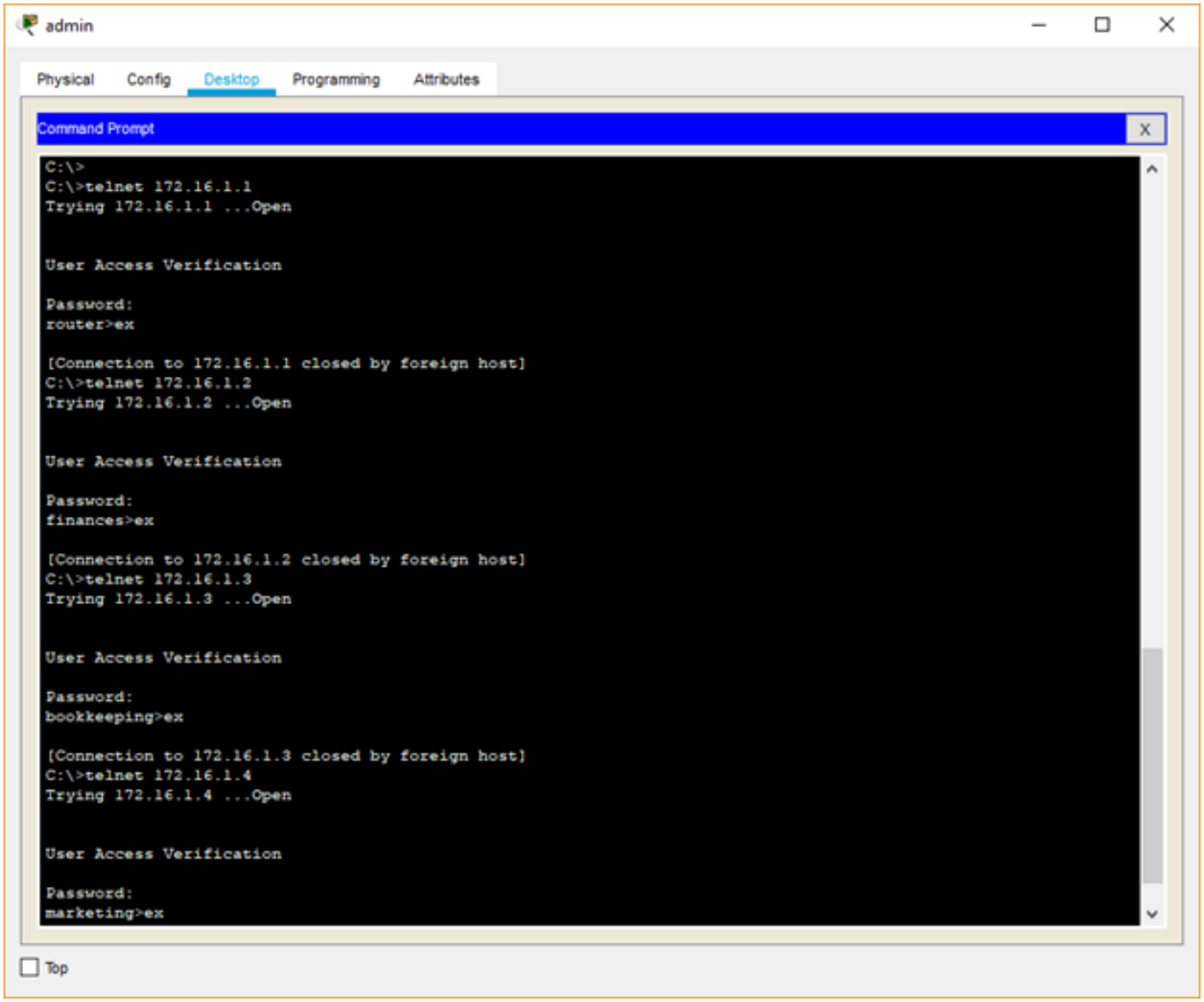

Рис. 5.6.5 - Результат команд telnet на комп'ютері admin.

Розділ 6. Робота користувача з програмною системою

При запуску exe файла перед користувачем відкривається панель редагування рис. 7

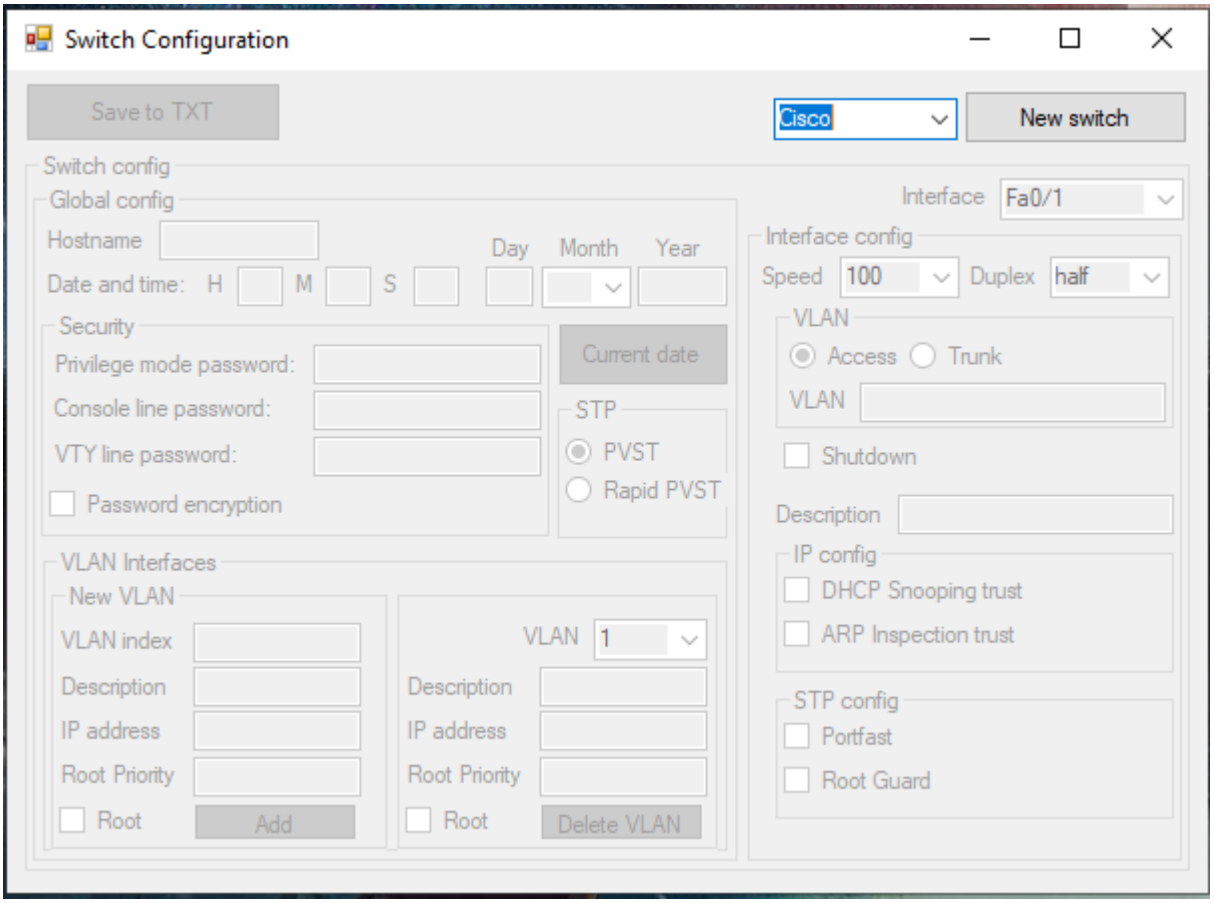

Рис. 6.1 – Панель редагування конфігурації

Користувачу для початку роботи необхідно обрати модель комутатора для редагування налаштувань.(Наразі доступний тільки cisco)

Коли модель комутатора обрано, необхідно натиснути “New switch”, тоді відкриваэться доступ до налаштувань.

Спершу, заповнюємо поля для Глобальної конфігурації у розділі “Global config”. Вказуємо ім’я хосту, дату, налаштовуємо паролі, обираємо тип STP та підключаємо шифрування паролів по бажанню. Рис.8

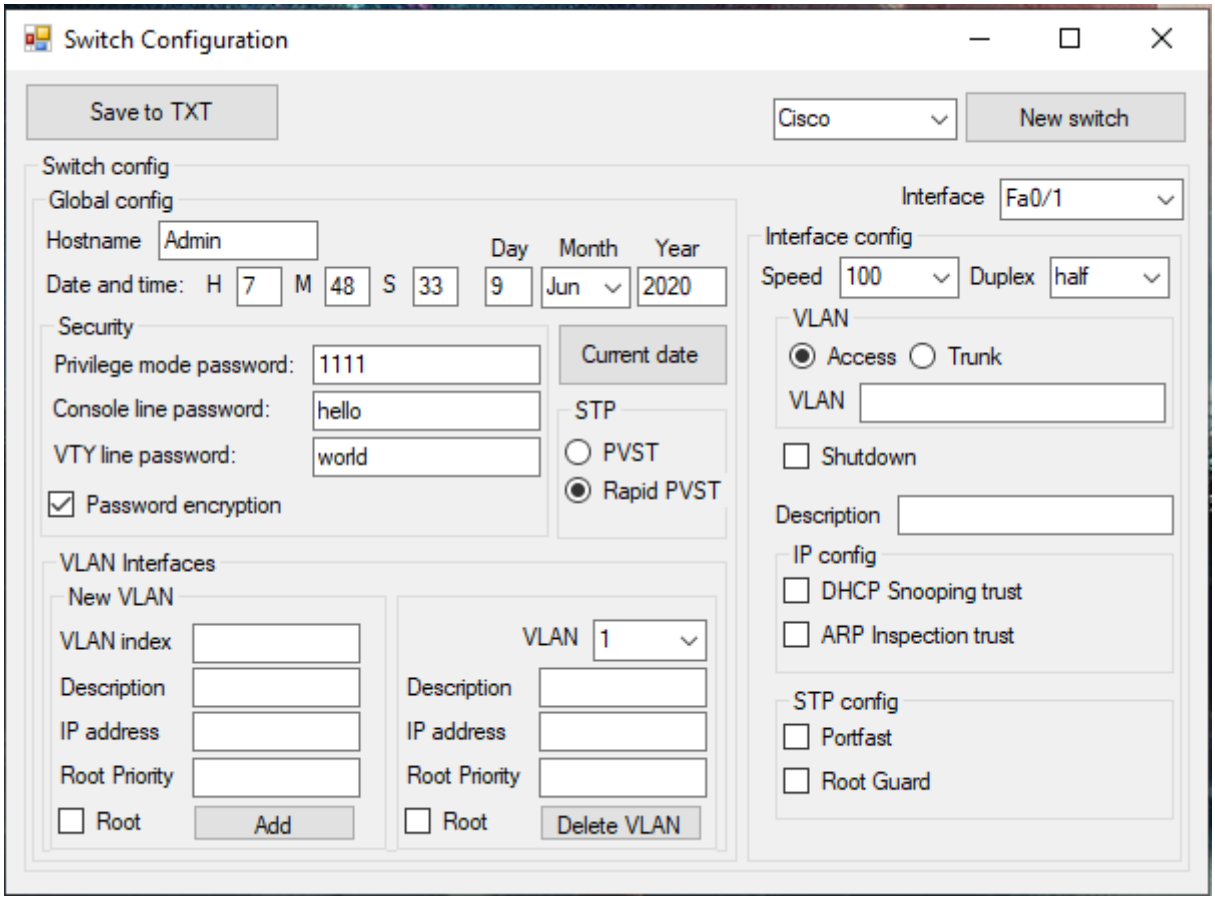

Рис. 6.2 – Налаштування Global config

Далі користувач повинен створити VLAN, для цього потрібно у розділі “New VLAN” вказати номер влану від 2 до 4096, опис влану, ip адресу комутатора у цьому влані, та рівень доступу. При необхідності, можна одрузу позначити влан як корінний Рис. 8

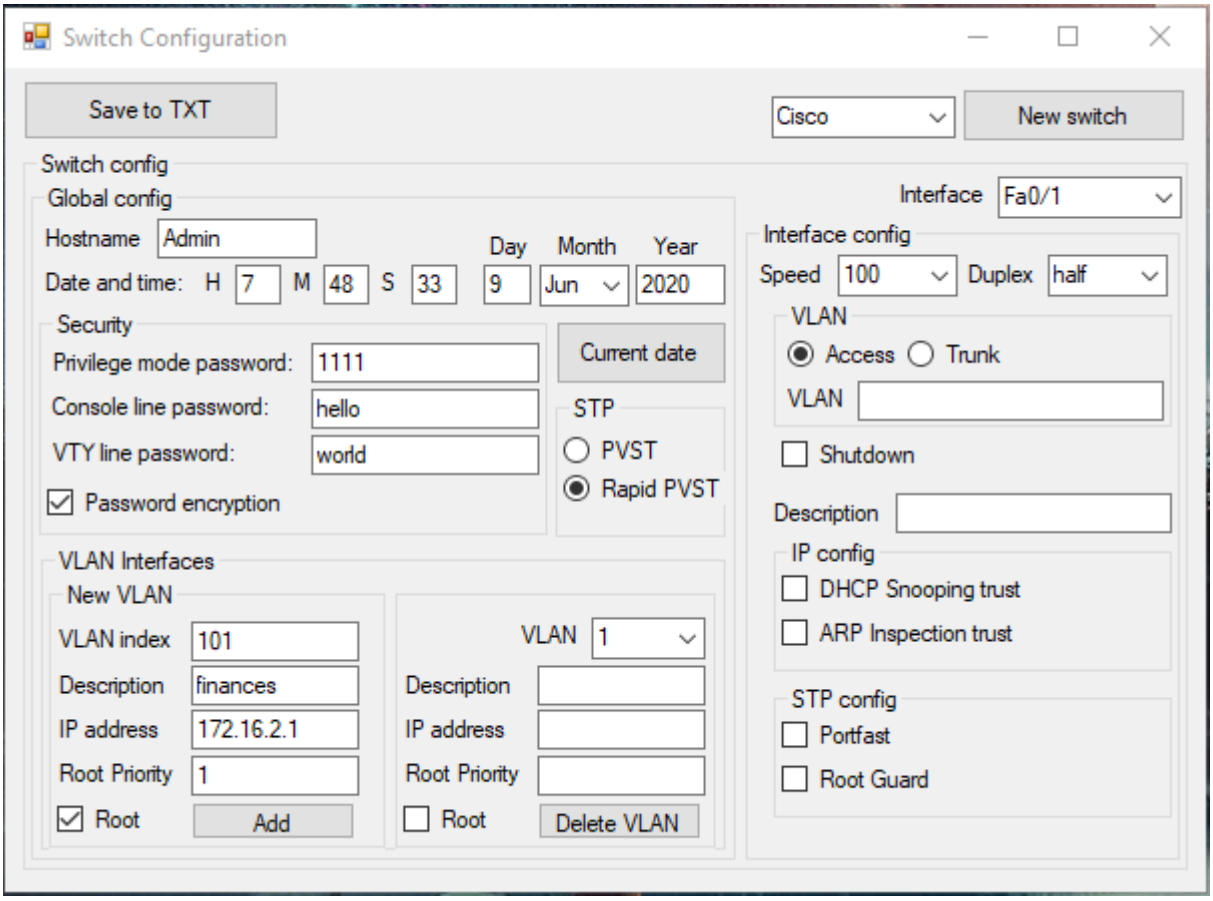

Рис. 6.3 – Налаштування “New VLAN”

Після заповнення інформації по новому влану натискаємо кнопку “Add” для його збереження. Якщо при цьому ви ввели невірно номер влану або ip, виникне відповідна помилка Рис. 9

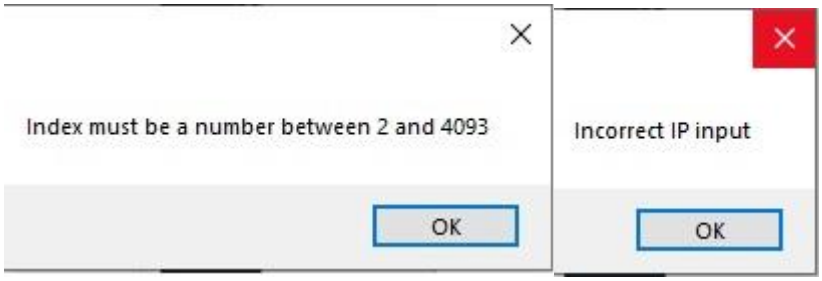

Рис. 6.4 та 6.5 – Помилки

Якщо користувач ввів коректні дані, влан буде збереженно. Натиснувши на випадаючий список VLAN, можна переглянути створенні влани і за бажанням редагувати їх, або видалити натиснувши кнопку “Delete VLAN”.

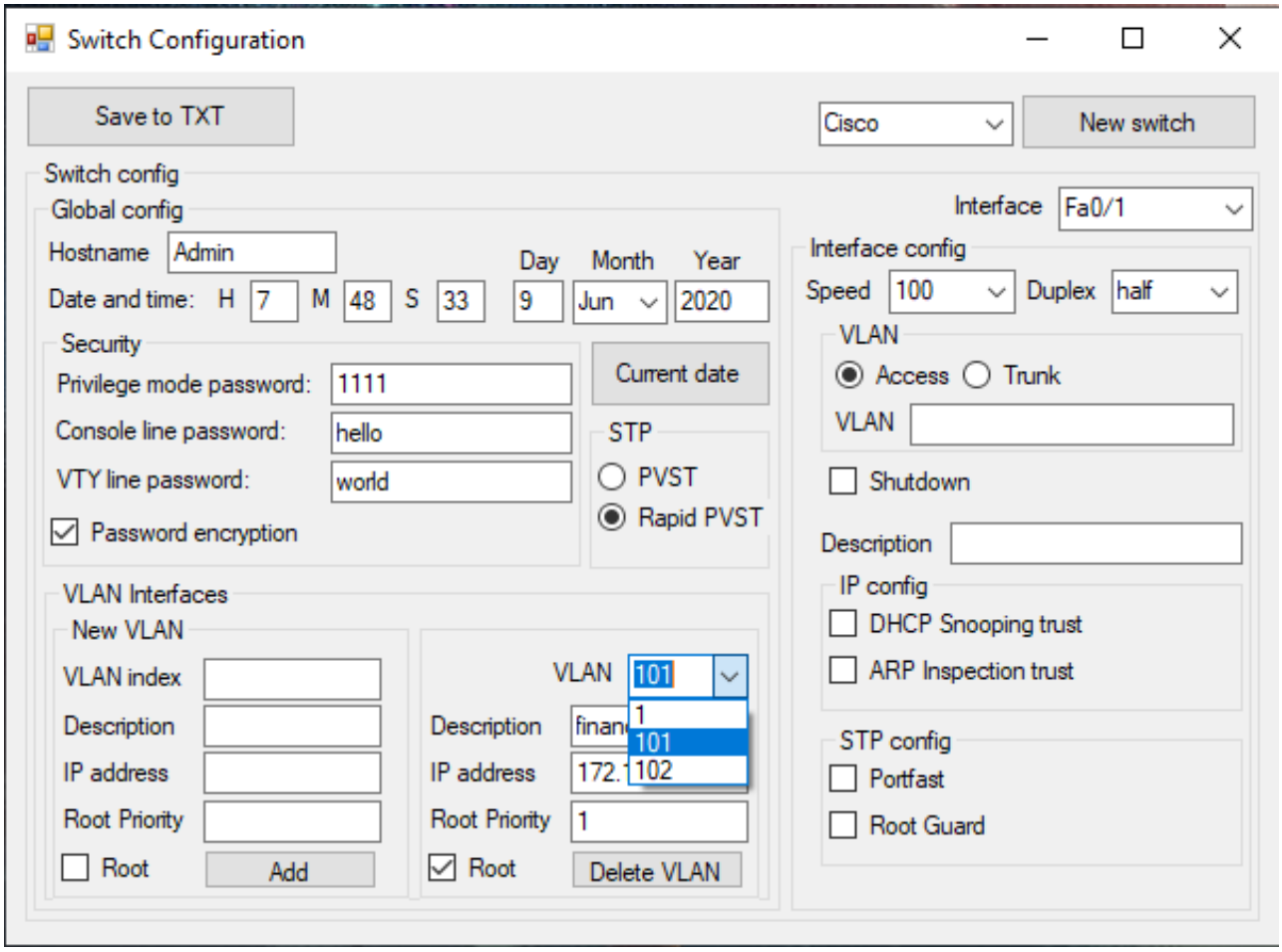

Рис. 6.6 – редагування, створенного влану.

Останніми користувач налаштовує порты. На правій частині панелі, у випадаючому списку “Interface”, обираємо по черзі кожний порт і вказуємо для нього необхідні налаштування у розділі “Interface config”. Для кожного порта окремо можна обрати швидкість, дуплекс, прив’язати до влану, визначити режим роботи, додати опис, підключити захист через IP config, вимкнути обмеження STP, а також підключити Root Guard, рис.10

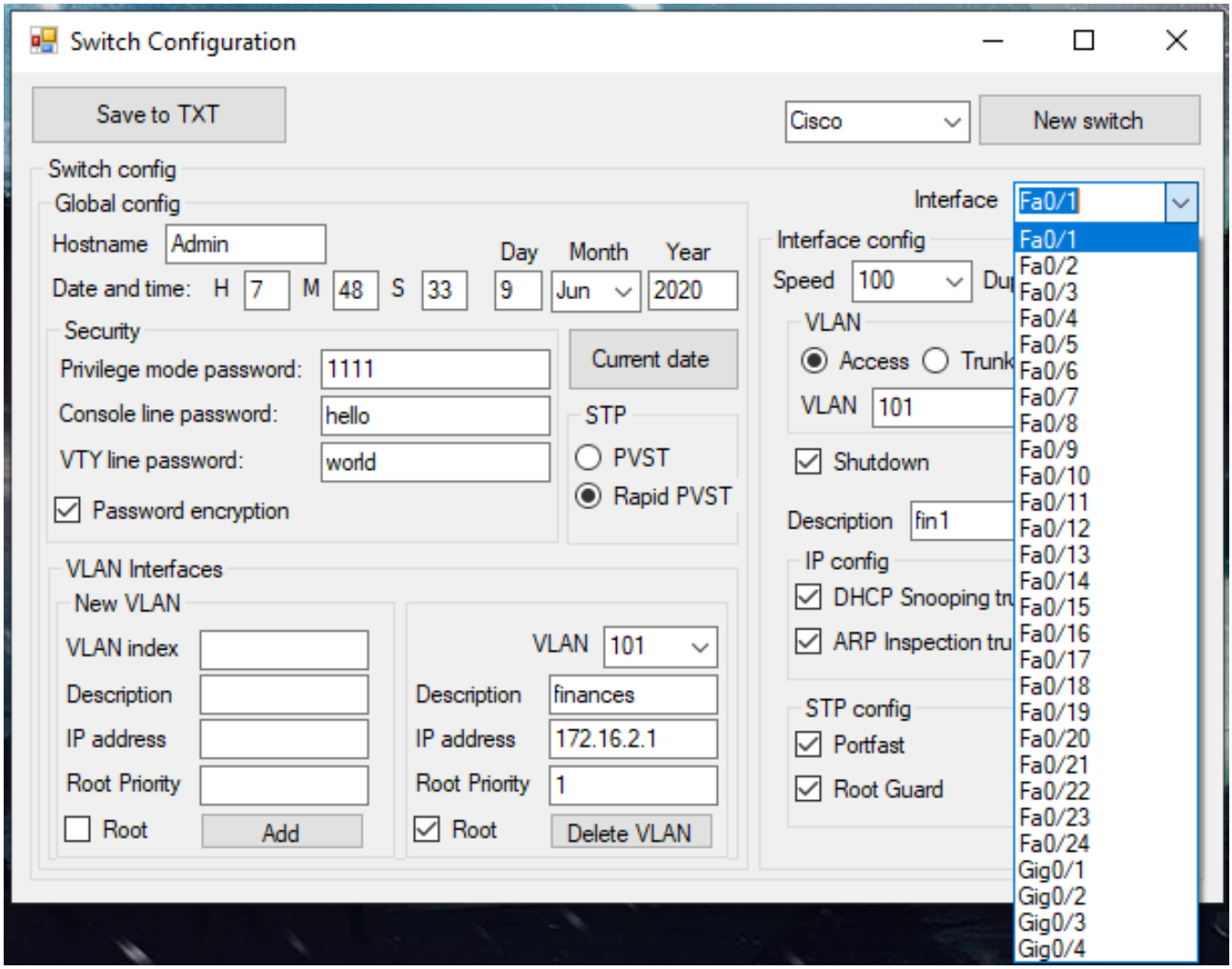

Рис. 6.7 – список портів для конфігурації

Коли користувач заповнив усі необхідні данні для налаштування комутатора, то натискає на “Save to TXT” і зберігає документ зі створеною конфігурацією.

Конфігурація буде матиме вигляд частково зображений на рис. 10

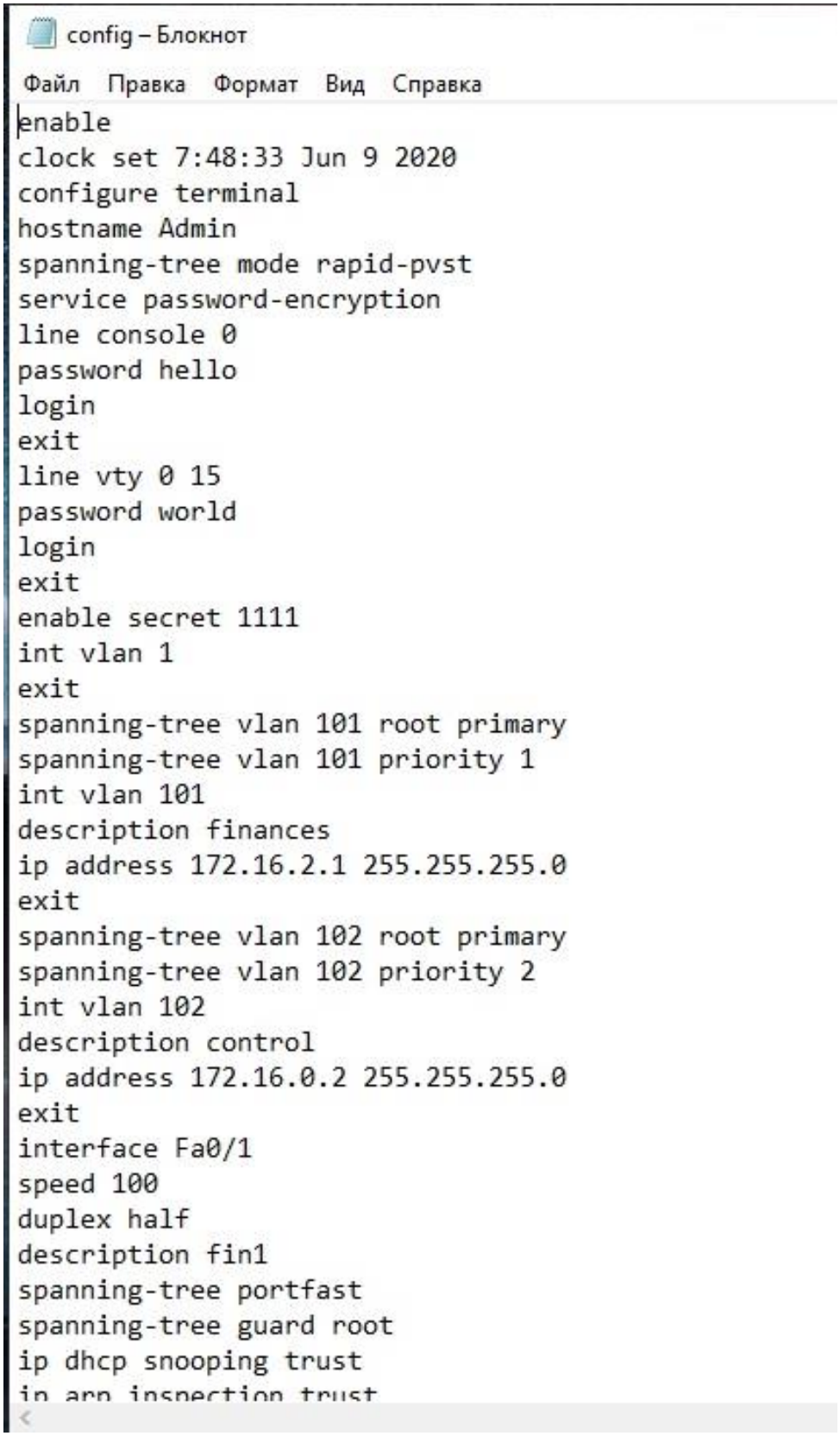

Рис.6.8 – частина документу з конфігурацією

Все, що залишилось зробити, це скопіювати створену конфігурацію на інтерфейс комутатора рис.11

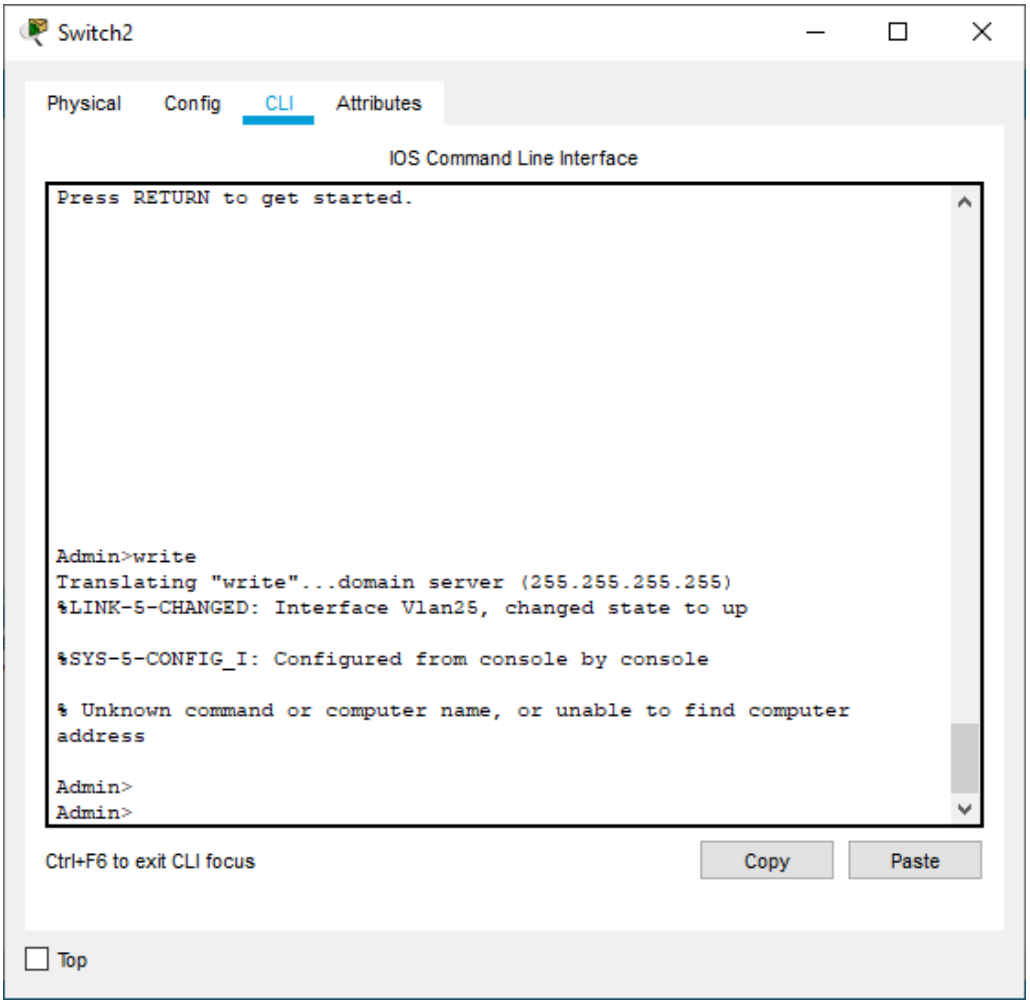

Рис.6.9 – інтерфейс комутатора cisco

Висновки

У результаті проведеної роботи, було створено систему автоматизованого налаштування комутаторів cisco, а також побудовано локальну мережу для підприємства з розмежованим доступом. Завдяки створеній програмі, часу на налаштування комутаторів було втрачено мінімум.

У наш час існує дуже багато різних моделей комутаторів. Майже кожна модель відрізняється від аналогів своїми командами.

Відпрацювавши деякий час в компанії інтернет-провайдеру, я зрозумів, що за один день робітникам відділу моніторингу доводиться пере-налаштовувати величезну кількість мережевого обладнання. Якщо в компанії буде програмне забезпечення яке містить у собі конфігурацію всіх комутаторів, то це значно оптимізує процес і зменшить рівень завантаженості працівників.

Мій проект передбачає подальшу підтримку цієї мультисервісної системи. До програми будуть додаватись дані про конфігурації інших моделей, таких як ZTE, DLINK, RAISCOM, Edjecore, Foxgate, Huawei, Linksys, BDCOM. Сподіваюсь, що у майбутньому моя система принесе користь багатьом компаніям.

Список використаних джерел

1. Видео курс от Андрея Созыкина по компьютерным наукам и информационным технологиям. 2018 - 2020

2. Д. Куроуз, К. Росс "Комп'ютерні мережі. Спадний підхід" (2016)

3. Майо, Джо. Самоучитель Microsoft Visual Studio 2010 : пер. с англ. / Дж. Майо. - Санкт-Петербург : БХВ-Петербург, 2011. - 464 с.

4. У. Одом "Официальное руководство Cisco по подготовке к сертификационным экзаменам CCNA ICND2 200-101. Маршрутизация и коммутация" (2016)

ДОДАТОК 1

Автоматизована система налаштування та супроводу мультисервісних комп’ютерних мереж

Специфікація

УКР.НТУУ “КПІ ім.І.Сікорського”.ТР-62153

Аркушів 2

2020

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ “КПІ ім.І.Сікорського”.	Записка.docx	Пояснювальна записка
Комплекс		
Компоненти		
УКР.НТУУ “КПІ ім.І.Сікорського”.	Form1.cs	Програмний модуль
УКР.НТУУ “КПІ ім.І.Сікорського”.	Form1.Designer.cs	Програмний модуль
УКР.НТУУ “КПІ ім.І.Сікорського”.	Port.cs	Програмний модуль
УКР.НТУУ “КПІ ім.І.Сікорського”.	Program.cs	Програмний модуль
УКР.НТУУ “КПІ ім.І.Сікорського”.	Switch.cs	Програмний модуль
УКР.НТУУ “КПІ ім.І.Сікорського”.	VLAN.cs	Програмний модуль
УКР.НТУУ “КПІ ім.І.Сікорського”.	SwitchConfiger.exe	Виконуваний файл інтерфейсу користувача
УКР.НТУУ “КПІ ім.І.Сікорського”.	LocalNet.pkt	Модель локальної мережі

ДОДАТОК 2

Автоматизована система налаштування та супроводу мультисервісних комп’ютерних мереж

Текст програмного модуля

УКР.НТУУ “КПІ ім.І.Сікорського”.ТР-62153

Аркушів 11

2020

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace SwitchConfiger
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        Switch Sw;
        bool changing;
        private void Form1_Load(object sender, EventArgs e)
        {
            changing = true;
            Sw = new Switch();
            comboVendor.SelectedIndex = 0;
            comboSpeed.SelectedIndex = 1;
            comboDuplex.SelectedIndex = 0;
            comboPort.SelectedIndex = 0;
            comboVlan.SelectedIndex = 0;
            changing = false;
        }

        private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (!changing)
            {
                Sw.Ports[comboPort.SelectedIndex].Speed = comboSpeed.SelectedItem.ToString();
            }
        }

        private void label4_Click(object sender, EventArgs e)
        {

        }

        private void groupBox6_Enter(object sender, EventArgs e)
        {

        }

        private void label6_Click(object sender, EventArgs e)
        {

        }

        private void textBox3_TextChanged(object sender, EventArgs e)
        {
            Sw.ConsolePas = textConsole.Text;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            groupBox1.Enabled = true;
            buttonSave.Enabled = true;
            Sw = new Switch();
            radioAccess.Checked = true;
            radioPVST.Checked = true;
            textPortVlan.Text = "";
```

```
    textNewDesc.Text = "";
    textNewIndex.Text = "";
    textNewIP.Text = "";
    textNewPrior.Text = "";
    textVlanDesc.Text = "";
    textVlanIP.Text = "";
    textVlanPrior.Text = "";
    textConsole.Text = "";
    textVTY.Text = "";
    textEnable.Text = "";
    textHostname.Text = "";
    checkARP.Checked = false;
    checkRoot.Checked = false;
    checkShutdown.Checked = false;
    checkSnooping.Checked = false;
    checkPortfast.Checked = false;
    checkNewRoot.Checked = false;
    checkVlanRoot.Checked = false;
    checkEncrypt.Checked = false;
    comboVendor.SelectedIndex = 0;
    comboSpeed.SelectedIndex = 1;
    comboDuplex.SelectedIndex = 0;
    comboPort.SelectedIndex = 0;
    comboVlan.SelectedIndex = 0;
}

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    if(comboVendor.SelectedIndex != 0)
    {
        MessageBox.Show("Not implemented");
        comboVendor.SelectedIndex = 0;
    }
}

private void comboBox4_SelectedIndexChanged(object sender, EventArgs e) //interface
{
    changing = true;
    Port currentPort = Sw.Ports[comboPort.SelectedIndex];
    if (comboPort.SelectedIndex < 24 && comboSpeed.Items.Count == 4)
    {
        comboSpeed.Items.Clear();
        comboSpeed.Items.Add("10");
        comboSpeed.Items.Add("100");
        comboSpeed.Items.Add("auto");

    }
    if (comboPort.SelectedIndex >= 24 && comboSpeed.Items.Count == 3)
    {
        comboSpeed.Items.Clear();
        comboSpeed.Items.Add("10");
        comboSpeed.Items.Add("100");
        comboSpeed.Items.Add("1000");
        comboSpeed.Items.Add("auto");
    }

    if(comboPort.SelectedIndex < 24)
    {
        switch (currentPort.Speed)
        {
            case "10":
                comboSpeed.SelectedIndex = 0;
                break;
            case "100":
                comboSpeed.SelectedIndex = 1;
                break;
            case "auto":
                comboSpeed.SelectedIndex = 2;
                break;
```

```
        }
    }
    else
    {
        switch (currentPort.Speed)
        {
            case "10":
                comboSpeed.SelectedIndex = 0;
                break;
            case "100":
                comboSpeed.SelectedIndex = 1;
                break;
            case "1000":
                comboSpeed.SelectedIndex = 2;
                break;
            case "auto":
                comboSpeed.SelectedIndex = 3;
                break;
        }
    }

    switch(currentPort.Duplex)
    {
        case "half":
            comboDuplex.SelectedIndex = 0;
            break;
        case "full":
            comboDuplex.SelectedIndex = 1;
            break;
        case "auto":
            comboDuplex.SelectedIndex = 2;
            break;
    }
    if (currentPort.Switchport == "trunk")
        radioTrunk.Checked = true;
    else
        radioAccess.Checked = true;
    textPortVlan.Text = currentPort.VLAN;
    textDesc.Text = currentPort.Description;
    if (currentPort.Admin == "shutdown")
        checkShutdown.Checked = true;
    else
        checkShutdown.Checked = false;


    if (currentPort.Snooping == "ip dhcp snooping trust")
        checkSnooping.Checked = true;
    else
        checkSnooping.Checked = false;

    if (currentPort.ARPinspection == "ip arp inspection trust")
        checkARP.Checked = true;
    else
        checkARP.Checked = false;

    if (currentPort.Portfast == "spanning-tree portfast")
        checkPortfast.Checked = true;
    else
        checkPortfast.Checked = false;

    if (currentPort.RootGuard == "spanning-tree guard root")
        checkRoot.Checked = true;
    else
        checkRoot.Checked = false;
    changing = false;
}

private void comboDuplex_SelectedIndexChanged(object sender, EventArgs e)
{
```

```
    if (!changing)
    {
        Sw.Ports[comboPort.SelectedIndex].Duplex = comboDuplex.SelectedItem.ToString();
    }
}

private void radioAccess_CheckedChanged(object sender, EventArgs e)
{
    if (!changing)
    {
        if (radioAccess.Checked)
            Sw.Ports[comboPort.SelectedIndex].Switchport = "access";
    }
}

private void radioTrunk_CheckedChanged(object sender, EventArgs e)
{
    if (!changing)
    {
        if (radioTrunk.Checked)
            Sw.Ports[comboPort.SelectedIndex].Switchport = "trunk";
    }
}

private void textPortVlan_TextChanged(object sender, EventArgs e)
{
    if (!changing)
    {
        Sw.Ports[comboPort.SelectedIndex].VLAN = textPortVlan.Text;
    }
}

private void checkShutdown_CheckedChanged(object sender, EventArgs e)
{
    if (!changing)
    {
        if (checkShutdown.Checked)
            Sw.Ports[comboPort.SelectedIndex].Admin = "shutdown";
        else
            Sw.Ports[comboPort.SelectedIndex].Admin = "no shutdown";
    }
}

private void checkSnooping_CheckedChanged(object sender, EventArgs e)
{
    if (!changing)
    {
        if (checkSnooping.Checked)
            Sw.Ports[comboPort.SelectedIndex].Snooping = "ip dhcp snooping trust";
        else
            Sw.Ports[comboPort.SelectedIndex].Snooping = "no ip dhcp snooping trust";
    }
}

private void checkARP_CheckedChanged(object sender, EventArgs e)
{
    if (!changing)
    {
        if (checkARP.Checked)
            Sw.Ports[comboPort.SelectedIndex].ARPinspection = "ip arp inspection trust";
        else
            Sw.Ports[comboPort.SelectedIndex].ARPinspection = "no ip arp inspection trust";
    }
}

private void checkPortfast_CheckedChanged(object sender, EventArgs e)
{
    if (!changing)
    {
```

```
            if (checkPortfast.Checked)
                Sw.Ports[comboPort.SelectedIndex].Portfast = "spanning-tree portfast";
            else
                Sw.Ports[comboPort.SelectedIndex].Portfast = "no spanning-tree portfast";
        }
    }

    private void checkRoot_CheckedChanged(object sender, EventArgs e)
    {
        if (!changing)
        {
            if (checkRoot.Checked)
                Sw.Ports[comboPort.SelectedIndex].RootGuard = "spanning-tree guard root";
            else
                Sw.Ports[comboPort.SelectedIndex].RootGuard = "no spanning-tree guard root";
        }
    }

    private void textEnable_TextChanged(object sender, EventArgs e)
    {
        Sw.EnablePas = textEnable.Text;
    }

    private void textVTY_TextChanged(object sender, EventArgs e)
    {
        Sw.VTYPas = textVTY.Text;
    }

    private void checkEncrypt_CheckedChanged(object sender, EventArgs e)
    {
        Sw.Encryption = checkEncrypt.Checked;
    }

    private void radioPVST_CheckedChanged(object sender, EventArgs e)
    {
        if (radioPVST.Checked)
            Sw.STP = "spanning-tree mode pvst";
        else
            Sw.STP = "spanning-tree mode rapid-pvst";
    }

    private void buttonAddVlan_Click(object sender, EventArgs e)
    {
        bool repeat = false;
        for (int i = 0; i < Sw.Vlans.Count; i++)
            if (Sw.Vlans[i].Index == textNewIndex.Text)
                repeat = true;
        if(repeat)
        {
            MessageBox.Show("VLAN index already exists.");
            textNewIndex.Text = "";
        }
        else
        {
            if(Int32.TryParse(textNewIndex.Text, out int index) && index > 1 && index < 4094)
            {
                Regex lookForIp = new Regex(@"\b(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-
4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\b");
                if (textNewIP.Text == "" || lookForIp.IsMatch(textNewIP.Text))
                {
                    if (textNewPrior.Text == "" || Int32.TryParse(textNewPrior.Text, out _))
                    {
                        VLAN newVlan = new VLAN(index)
                        {
                            Description = textNewDesc.Text,
                            IP = textNewIP.Text,
                            Priority = textNewPrior.Text
                        };
                        if (checkNewRoot.Checked)
```

```
                newVlan.Root = $"spanning-tree vlan {textNewIndex.Text} root primary";
            textNewDesc.Text = "";
            textNewPrior.Text = "";
            textNewIP.Text = "";
            textNewIndex.Text = "";
            checkNewRoot.Checked = false;
            Sw.Vlans.Add(newVlan);
            comboVlan.Items.Add(newVlan.Index);
         }
         else
         {
            MessageBox.Show("Root priority must be a number");
            textNewPrior.Text = "";
         }
      }
      else
      {
         MessageBox.Show("Incorrect IP input");
         textNewIP.Text = "";
      }
   }
   else
   {
      MessageBox.Show("Index must be a number between 2 and 4093");
      textNewIndex.Text = "";
   }
}

private void comboVlan_SelectedIndexChanged(object sender, EventArgs e)
{
   changing = true;
   VLAN currentVlan = Sw.Vlans[comboVlan.SelectedIndex];
   textVlanDesc.Text = currentVlan.Description;
   textVlanIP.Text = currentVlan.IP;
   textVlanPrior.Text = currentVlan.Priority;
   checkVlanRoot.Checked = !(currentVlan.Root == "");
   changing = false;
}

private void button1_Click_1(object sender, EventArgs e)
{
   if (comboVlan.SelectedIndex == 0)
   {
      MessageBox.Show("Cannot delete 1 VLAN");
   }
   else
   {
      int i = comboVlan.SelectedIndex;
      comboVlan.SelectedIndex = 0;
      comboVlan.Items.RemoveAt(i);
      Sw.Vlans.RemoveAt(i);
   }
}

private void textVlanDesc_TextChanged(object sender, EventArgs e)
{
   if (!changing)
   {
      Sw.Vlans[comboVlan.SelectedIndex].Description = textVlanDesc.Text;
   }
}

private void textVlanIP_TextChanged(object sender, EventArgs e)
{
   if (!changing)
   {
      Sw.Vlans[comboVlan.SelectedIndex].IP = textVlanIP.Text;
   }
```

```
}

private void textVlanPrior_TextChanged(object sender, EventArgs e)
{
   if (!changing)
   {
      if (textNewPrior.Text == "" || Int32.TryParse(textNewPrior.Text, out _))
      {
         Sw.Vlans[comboVlan.SelectedIndex].Priority = textVlanPrior.Text;
      }
      else
         MessageBox.Show("Root priority must be a number");
   }
}

private void buttonSave_Click(object sender, EventArgs e)
{
   SaveFileDialog f = new SaveFileDialog();
   f.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";
   if (f.ShowDialog() == DialogResult.OK)
   {
      Sw.Save(f.FileName);
   }
}

private void textVlanIP_ModifiedChanged(object sender, EventArgs e)
{

}

private void textPortVlan_Leave(object sender, EventArgs e)
{
}

private void textBox1_TextChanged(object sender, EventArgs e)
{
   Sw.Hostname = textHostname.Text;
}

private void textNewPrior_TextChanged(object sender, EventArgs e)
{

}

private void textDesc_TextChanged(object sender, EventArgs e)
{
   Sw.Ports[comboPort.SelectedIndex].Description = textDesc.Text;
}

private void buttonDate_Click(object sender, EventArgs e)
{
   DateTime now = DateTime.Now;
   textHour.Text = now.Hour.ToString();
   textMin.Text = now.Minute.ToString();
   textSec.Text = now.Second.ToString();
   textDay.Text = now.Day.ToString();
   textYear.Text = now.Year.ToString();
   comboMonth.SelectedIndex = now.Month - 1;
}

private void textHour_TextChanged(object sender, EventArgs e)
{
   Sw.Hour = textHour.Text;
}

private void textMin_TextChanged(object sender, EventArgs e)
{
   Sw.Min = textMin.Text;
}
```

```
        private void textSec_TextChanged(object sender, EventArgs e)
        {
            Sw.Sec = textSec.Text;
        }

        private void textDay_TextChanged(object sender, EventArgs e)
        {
            Sw.Day = textDay.Text;
        }

        private void comboMonth_SelectedIndexChanged(object sender, EventArgs e)
        {
            Sw.Month = comboMonth.SelectedItem.ToString();
        }

        private void textYear_TextChanged(object sender, EventArgs e)
        {
            Sw.Year = textYear.Text;
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SwitchConfiger
{
    public class Port
    {
        public string Speed { get; set; }
        public string Duplex { get; set; }
        public string VLAN { get; set; }
        public string Switchport { get; set; }
        public string Portfast { get; set; }
        public string RootGuard { get; set; }
        public string Snooping { get; set; }
        public string ARPinspection { get; set; }
        public string Admin { get; set; }
        public string Name { get; set; }
        public string Description { get; set; }

        public Port(int index)
        {
            if (index < 24)
            {
                Name = "Fa0/" + (index + 1);
            }
            else
            {
                Name = "Gig0/" + (index - 23);
            }
            Speed = "100";
            Duplex = "half";
            VLAN = "";
            Switchport = "access";
            Portfast = "";
            RootGuard = "";
            Snooping = "";
            ARPinspection = "";
```

```
            Admin = "no shut";
        }


    }
}


namespace SwitchConfiger
{
    public class Switch
    {
        public List<Port> Ports = new List<Port>();
        public List<VLAN> Vlans = new List<VLAN>();
        public string ConsolePas { get; set; }
        public string Hostname { get; set; }
        public string EnablePas { get; set; }
        public string VTYPas { get; set; }
        public bool Encryption { get; set; }
        public string STP { get; set; }
        public string Min { get; set; }
        public string Sec { get; set; }
        public string Hour { get; set; }
        public string Day { get; set; }
        public string Month { get; set; }
        public string Year { get; set; }
        public Switch()
        {
            STP = "";
            ConsolePas = "";
            EnablePas = "";
            VTYPas = "";
            Encryption = false;
            for (int i = 0; i < 28; i++)
            {
                Ports.Add(new Port(i));
            }
            Vlans.Add(new VLAN(1));
        }

        public void Save(string path)
        {
            List<string> commands = new List<string>();
            commands.Add("enable");
            commands.Add($"clock set {Hour}:{Min}:{Sec} {Month} {Day} {Year}");
            commands.Add("configure terminal");
            if (Hostname != "")
                commands.Add("hostname " + Hostname);
            if (STP != "")
                commands.Add(STP);
            if (Encryption)
                commands.Add("service password-encryption");
            if(ConsolePas != "")
            {
                commands.Add("line console 0");
                commands.Add("password " + ConsolePas);
                commands.Add("login");
                commands.Add("exit");
            }
            if (VTYPas != "")
            {
                commands.Add("line vty 0 15");
                commands.Add("password " + VTYPas);
                commands.Add("login");
                commands.Add("exit");
            }
            if(EnablePas != "")
                commands.Add("enable secret " + EnablePas);
            for(int i = 0; i < Vlans.Count; i++)
```

```
            {
                if(Vlans[i].Root != "")
                    commands.Add(Vlans[i].Root);
                if (Vlans[i].Priority != "")
                    commands.Add($"spanning-tree vlan {Vlans[i].Index} priority {Vlans[i].Priority}");
                commands.Add("int vlan " + Vlans[i].Index);
                if (Vlans[i].Description != "")
                    commands.Add("description " + Vlans[i].Description);
                if (Vlans[i].IP != "")
                    commands.Add("ip address " + Vlans[i].IP + " 255.255.255.0");
                commands.Add("exit");
            }
            for (int i = 0; i < Ports.Count; i++)
            {
                commands.Add("interface " + Ports[i].Name);
                commands.Add("speed " + Ports[i].Speed);
                commands.Add("duplex " + Ports[i].Duplex);
                commands.Add("description " + Ports[i].Description);
                commands.Add(Ports[i].Portfast);
                commands.Add(Ports[i].RootGuard);
                commands.Add(Ports[i].Snooping);
                commands.Add(Ports[i].ARPinspection);
                commands.Add(Ports[i].Admin);
                commands.Add("switchport mode " + Ports[i].Switchport);
                if(Ports[i].Switchport == "trunk")
                {
                    commands.Add("switchport trunk allowed vlan " + Ports[i].VLAN);
                }
                else
                {
                    commands.Add("switchport access vlan " + Ports[i].VLAN);
                }
                commands.Add("exit");
            }
            commands.Add("exit");
            commands.Add("write");
            File.WriteAllLines(path, commands.ToArray());
        }
    }
}


namespace SwitchConfiger
{
    public class VLAN
    {
        public string Description { get; set; }
        public string Index { get; set; }
        public string IP { get; set; }
        public string Root { get; set; }
        public string Priority { get; set; }

        public VLAN(int index)
        {
            Description = "";
            Index = index.ToString();
            IP = "";
            Root = "";
            Priority = "";
        }
    }
}
```

ДОДАТОК 3

Автоматизована система налаштування та супроводу мультисервісних комп’ютерних мереж

Опис програмного модуля

УКР.НТУУ “КПІ ім.І.Сікорського”.

Аркушів 6

2020

ЗМІСТ

1 ЗАГАЛЬНІ ВІДОМОСТІ

Назва програмного продукту демонстрації – “ SwitchConfiger”. Мова програмної реалізації – C++. Середовище розробки – Microsoft Visual Studio 2019.

Для демонстраціїї ефективності розробленої системи необхідно:

- комп’ютер або ноутбук;
- операційна система Windows;
- мережевий комутатор моделі cisco;

2 ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розроблена система являє собою програмний продук за допомогою якого користувач зможе значно пришвидшити процесс конфігурування мережевих комутаторів.

Проект, зможе досягти значних успіхів у випадку його поширення на нові моделі комутаторів. У цьому зможе переконатись будь-яка інтернет компанія, яка працює з великою кідькістю різноманітного мережевого обладнання.

3 ВХІДНІ ДАНІ

Вхідними даними нашої програми можна вважати зчитавання заповнених користувачем відповідних розділів у інтерфейсі нашого програмному продукту.

Ось наприклад на рисунку 3.1 приклад заповнених полів.

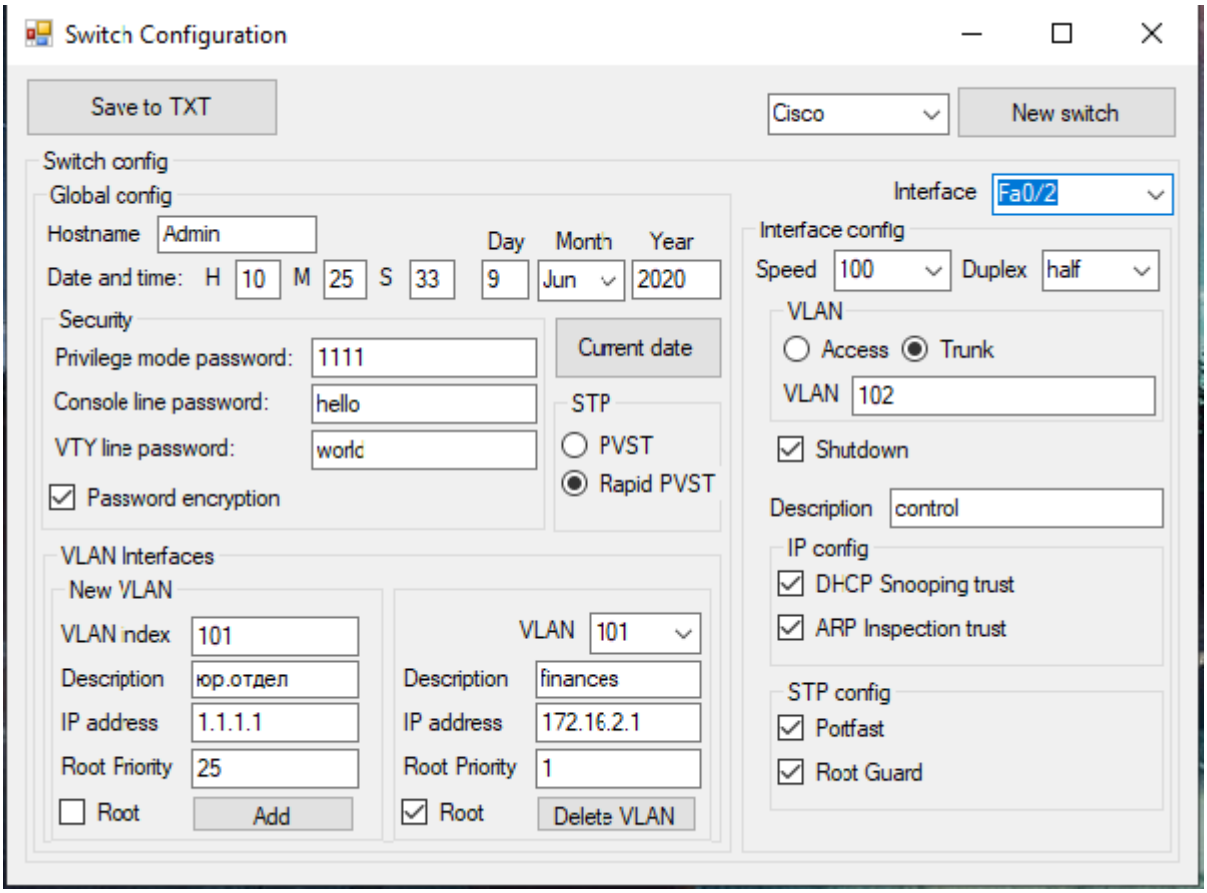

Рисунок 3.1 – Приклад вхідних даних на інтерфейсі користувача

Потім вся вказанна на полях інформація буде зчитана та перетворенна на зрозумілий для комутатора текст.

-6-

4 ВИХІДНІ ДАНІ

Вихідними даними є документ, в який записуються всі необхідні команди (рис 4.1) для створення відповідної конфігурації визначеної користувачем на інтерфейсі програми. Для конфігурації комутатора нам потрібно скопіювати згенеровані команди у інтерфейс комутатора.

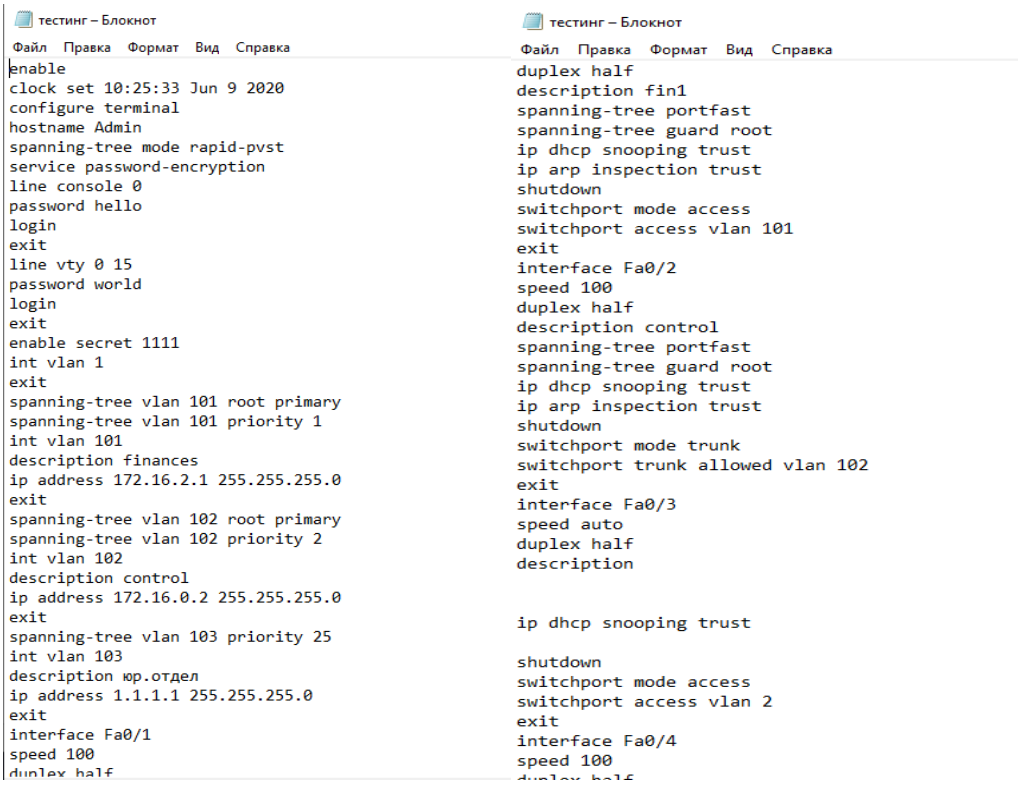

Рисунок 4.1 – Приклад вихідних даних